IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

APPLICANTS:	Venkatesh R. Iyer, Shilpa P. Bansod, and Senjeevan P. Raghavendran
SERIAL NO.:	10/039,111
FILING DATE:	January 4, 2002
TITLE:	System and Method for Determining Real-Time Application Verb Response Time
EXAMINER:	Jeffrey R. Swearingen
GROUP ART UNIT:	2145
ATTY. DKT. NO.:	24523-09627

MAIL STOP AMENDMENT
COMMISSIONER FOR PATENTS
P.O. BOX 1450
ALEXANDRIA, VA 22313-1450

DECLARATION OF VENKATESH R. IYER
UNDER 37 CFR 1.132

I, Venkatesh R. Iyer, hereby declare unequivocally the following:

1. I am a joint inventor of the above-referenced patent application.

2. This patent application relates to the field of network analysis.

3. Independent claims 31, 38, and 45 of this patent application include the term "application verb." Further, all dependent claims 32-37 and 39-44 of this patent application include the term "application verb" by virtue of their dependency from independent claims 31 and 38, respectively.

4. Attached as Exhibit A is a document entitled "Remote Monitoring MIB Extensions for Identifying Application Protocol Verbs." (Hereinafter referred to as "7-17-2001 INTERNET-DRAFT"). As shown on page 1 of the 7-17-2001 INTERNET-DRAFT, the 7-17-2001 INTERNET-DRAFT was published on July 17, 2001, prior to the filing date (January 4, 2002) of this patent application, on the website of the Internet Engineering Task Force (IETF). Exhibit A (7-17-2001 INTERNET-DRAFT), page 1. The 7-17-2001 INTERNET-DRAFT is currently available on the Internet at the URL: http://tools.ietf.org/html/draft-ietf-rmonmib-appverbs-02.

5. The IETF is an open standards organization that develops and promotes Internet standards. It includes an international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet.

6. Attached as Exhibit B is a document entitled "Internet-Drafts" obtained from the Internet at the URL http://www.ietf.org/ID.html, and attached as Exhibit C is a document entitled "Guidelines to Authors of Internet-Drafts" obtained from the Internet at URL http://www.ietf.org/ietf/1id-guidelines.html. Exhibits B and C contain general information about IETF Internet-Drafts.

7. Briefly, Internet-Drafts are working documents of the IETF, its areas, and its working groups. See Exhibit B, page 1, paragraph 1. Internet-Drafts provide authors with the ability to distribute and solicit comments on

documents they eventually submit for publication as a Request for Comments (RFC). See Exhibit C, pages 1-2, section 2, paragraph 1. RFCs are a series of memoranda describing Internet technologies. RFCs and Internet-Drafts are widely read and used by persons of ordinary skill in the field of computer networking including network analysis to which this patent application pertains.

8. In general, Internet-Drafts can be accessed on the Internet by anyone on the IETF website. For example, the distribution of the 7-17-2001 INTERNET-DRAFT was unlimited. See Exhibit A (7-17-2001 INTERNET-DRAFT), page 1 ("Distribution of this document is unlimited.").

9. The IETF website was known to and frequently accessed by persons of ordinary skill in the field of computer networking, including network analysis, prior to and as of the filing date of this patent application, January 4, 2002, and continues to be a widely known and frequently accessed website.

10. The 7-17-2001 INTERNET-DRAFT defines an "application verb" as:

> Application Verb: Also called simply 'verb'. Refers to one of potentially many protocol operations that are defined by a particular application protocol.
>
> Note that an application verb is not equivalent to an application protocol sub-command or opcode within a packet containing a PDU for the application. An application verb is a transaction type, and may involve several PDU types within the application protocol (e.g., SNMP Get-PDU and Response-PDU). In some applications, a

verb may encompass protocol operations pertaining to more than one protocol entry in the protocol directory (e.g., ftp and ftp-data).

See Exhibit A (7-17-2001 INTERNET DRAFT), page 5, section 5.3.

11. The 7-17-2001 INTERNET-DRAFT also provides several examples of application verbs for various protocols. See Exhibit A (7-17-2001 INTERNET-DRAFT), pages 14-18, section 8. For example, "user", "pass", and "acct" are provided as examples of application verbs for the File Transfer Protocol (FTP) application. See Exhibit A (7-17-2001 INTERNET DRAFT), page 14, section 8.1. For another example, see page 17, section 8.4, where "get", "head", and "post" are provided as examples of application verbs for the Hypertext Transfer Protocol (HTTP) application. Exhibit A (7-17-2001 INTERNET-DRAFT), page 17, section 8.4.

12. The definition of the term "application verb" and its examples provided in the 7-17-2001 INTERNET-DRAFT are consistent with the description of "application verb" provided in the specification of this patent application. For example, page 4, line 10 and page 9, line 1 of the specification mention that "an 'application verb' may include any specific application transaction or transaction type." Also, page 15, line 15 and page 17, lines 2-3 of the specification mention HTTP "get" as an example of an application verb.

13. The 7-17-2001 INTERNET-DRAFT was published on July 17, 2001 and expired on January 17, 2002 (Internet-Drafts expire six months after publication). This Internet Draft was preceded by versions published in July

2000 (attached as Exhibit D) and November 2000 (attached as Exhibit E), and followed by versions published in February 2002 and August 2002. The 7-17-2001 INTERNET-DRAFT was then incorporated into RFC 3395 entitled "Remote Network Monitoring MIB Protocol Identifier Reference Extensions," published in September 2002, currently available at ftp://ftp.rfc-editor.org/in-notes/rfc3395.txt. A copy of RFC 3395 is attached hereto as Exhibit F.

14. RFC 3395 provides a definition and examples of application verb consistent with the 7-17-2001 INTERNET-DRAFT and the specification of this patent application. See Exhibit F (RFC 3395), page 4, section 2.3. In addition, all previous and subsequent versions of the 7-17-2001 INTERNET-DRAFT also discuss and provide examples of application verbs consistent with the 7-17-2001 INTERNET-DRAFT and the specification of this patent application. See, for example, Exhibit D, page 5, section 5.3, and Exhibit E, page 5, section 5.3.

15. Therefore, I believe that, as of the filing date (January 4, 2002) of the above-referenced patent application, a person of ordinary skill in the field of network analysis would have known and understood what an "application verb" is.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or

imprisonment, or both, under section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Mar 13th 2008

Date

Venkatesh R. Iyer

Exhibit A

Internet Draft

Andy Bierman
Chris Bucci
Cisco Systems, Inc.
Russell Dietz
Apptitude, Inc.
Albin Warth
Netscout Systems, Inc.
17 July 2001

Remote Monitoring MIB Extensions for Identifying Application Protocol Verbs

<draft-ietf-rmonmib-appverbs-02.txt>

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026 [RFC2026].

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet- Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Distribution of this document is unlimited. Please send comments to the RMONMIB WG mailing list <rmonmib@ietf.org>.

Exhibit A

1. Copyright Notice

2. Abstract

This memo defines an experimental portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes the algorithms required to identify protocol operations (verbs) within the protocol encapsulations managed with the Remote Network Monitoring MIB Version 2 [RFC2021].

3. Table of Contents

Exhibit A

4. The SNMP Network Management Framework

The SNMP Management Framework presently consists of five major components:

- An overall architecture, described in RFC 2571 [RFC2571].
- Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIv1 and described in RFC 1155 [RFC1155], RFC 1212 [RFC1212] and RFC 1215 [RFC1215]. The second version, called SMIv2, is described in RFC 2578 [RFC2578], RFC 2579 [RFC2579] and RFC 2580 [RFC2580].
- Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in RFC 1157 [RFC1157]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in RFC 1901 [RFC1901] and RFC 1906 [RFC1906]. The third version of the message protocol is called SNMPv3 and described in RFC 1906 [RFC1906], RFC 2572 [RFC2572] and RFC 2574 [RFC2574].
- Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in RFC 1157 [RFC1157]. A second set of protocol operations and associated PDU formats is described in RFC 1905

[RFC1905].

- o A set of fundamental applications described in RFC 2573 [RFC2573] and the view-based access control mechanism described in RFC 2575 [RFC2575].

A more detailed introduction to the current SNMP Management Framework can be found in RFC 2570 [RFC2570].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the mechanisms defined in the SMI.

This memo does not specify a MIB module.

5. Overview

There is a need for a standardized way of identifying the protocol operations defined for particular application protocols. Different protocol operations can have very different performance characteristics, and it is desirable to collect certain metrics at this level of granularity. This memo defines extensions to the existing protocol identifier structure [RFC2895], and is intended to update, not obsolete, the existing protocol identifier encoding rules.

5.1. Protocol Identifier Framework

The RMON Protocol Identifier (PI) structure [RFC2895] allows for a variable number of layer identifiers. Each layer contributes 4 octets to the protocolDirID OCTET STRING and one octet to the protocolDirParameters OCTET STRING. These two MIB objects comprise the index into the protocolDirTable [RFC2021], and represent a globally unique identifier for a particular protocol encapsulation (or set of encapsulations if the wild-card base layer is used).

5.2. Protocol Identifier Extensions for Application Verbs

The existing RMON protocol identifier architecture requires that an application verb be represented by one additional protocol layer, appended to the protocol identifier for the parent application. Since some application verbs are defined as strings which can exceed 4 octets in length, an integer mapping must be provided for each string. This memo specifies how the verb layer is structured, as well as a verb identifier macro syntax for specification of verb name to integer mappings.

5.3. Terms

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119. [RFC2119]

This document uses some terms defined in the RMON Protocol Identifier Reference document [RFC2895], and some new terms that need introduction here.

Application Verb
: Also called simply 'verb'. Refers to one of potentially many protocol operations that are defined by a particular application protocol.

  Note that an application verb is not equivalent to an application protocol sub-command or opcode within a packet containing a PDU for the application. An application verb is a transaction type, and may involve several PDU types within the application protocol (e.g., SNMP Get-PDU and Response-PDU). In some applications, a verb may encompass protocol operations pertaining to more than one protocol entry in the protocol directory (e.g., ftp and ftp-data).

Connect Verb
: The special application verb associated with connection or session setup and tear-down traffic, and not attributed to any other verb for the application. This verb is assigned the enumeration value of zero, and the verb 'connect(0)' is implicitly defined for all application protocols.

Parent Application
: One of potentially many protocol encapsulations which identifies a particular application protocol. This term refers generically to any or all such encapsulations for a given set of application verbs.

Verb Layer
: The portion of the protocol identifier octet string which identifies the application verb.

Verb Set
: The group of verbs enumerated for a particular application protocol. The list of verb strings within a particular verb-identifier macro invocation is also called the verb set for that verb identifier.

5.4. Relationship to the RMON-2 MIB

The RMON-2 MIB [RFC2021] contains the protocolDirTable MIB objects used to identify all protocol encapsulations that can be monitored by a particular RMON agent.

This memo describes how these MIB objects are mapped by an implementation, for entries which identify application verbs. This document does not define any new MIB objects to identify application verbs.

5.5. Relationship to the RMON MIB Protocol Identifier Reference

The RMON MIB Protocol Identifier Reference [RFC2895] defines the RMON Protocol Identifier Macro Specification Language, as well as the encoding rules for the ProtocolDirID and protocolDirParameters OCTET STRINGs.

This memo defines extensions to the Protocol Identifier Reference document for the identification of application verb information. It does not obsolete any portion of the Protocol Identifier Reference document.

6. Definitions

6.1. Verb Identifier Macro Format

The following example is meant to introduce the verb-identifier macro. This macro-like construct is used to represent protocol verbs for a specific parent application.

6.1.1. Lexical Conventions

The following keyword is added to the PI language:

```
VERB-IDENTIFIER
```

6.1.2. Extended Grammar for the PI Language

The following is the extended BNF notation for the grammar with starting symbol <piFile>, for representing verb identifier macros. Note that only the term <piFile> is actually modified from the definition in [RFC2895]. The <piDefinition> syntax is not reproduced here, since this memo is intended to extend that definition, not replace it.

```
    -- a file containing one or more
    -- Protocol Identifier (PI) definitions
    <piFile> = [ <piDefinition> | <piVerbDefinition> ]...

    -- a PI definition
    <piVerbDefinition> =
      <parentProtoName> "VERB-IDENTIFIER"
            "DESCRIPTION" string
          [ "REFERENCE" string ]
            "::=" "{" <verbList> "}"

    -- a list of verb identifier string
    <verbList> = <verbId> [ <wspace> "," <wspace> <verbId> ]...

    -- a verb identifier string
    <verbId> = <verbName> [<wspace>] "(" [<wspace>]
              <verbEnum> [<wspace>] ")" [<wspace>]

    -- a verb name
    <verbName> = lcname

    -- a verb enumeration
    <verbEnum> = <posNum>

    -- a positive integer
    <posNum> = any integer value greater than zero and
               less than 16,777,216

    -- <piDefinition> syntax is defined in [RFC2895]
    -- <wspace> syntax is defined in [RFC2895]
    -- lcname syntax is defined in [RFC2895]
```

6.1.3. Mapping of the Parent Protocol Name

The "parentProtoName" value, called the "parent protocol name" MUST be an ASCII string consisting of 1 to 64 characters. The encoding rules are exactly as specified in section 6.2.4 of [RFC2895], for the mapping of the protocol name field. If a <protoName> and a <parentProtoName> field contain the same value, then they refer to the same protocol.

A protocol identifier macro SHOULD exist in the <piFile> for at least one encapsulation of the parent application protocol, if any verb identifier macros referencing that parent application are present in the

<piFile>.

6.1.4. Mapping of the DESCRIPTION Clause

The DESCRIPTION clause provides a textual description of the protocol verb set identified by this macro. It SHOULD NOT contain details about items covered by the DECODING and REFERENCE clauses. The DESCRIPTION clause MUST be present in all verb-identifier macro declarations.

6.1.5. Mapping of the REFERENCE Clause

If a publicly available reference document exists for this set of application protocol verbs, it SHOULD be listed here. Typically this will be a URL, otherwise it will be the name and address of the controlling body.

The REFERENCE clause is optional, but SHOULD be implemented if an authoritative reference exists which specifies the application protocol verbs defined in the <verbList> section of this macro.

6.1.6. Mapping of the Verb List Clause

The verb list clause MUST be present, and is used to identify a list of application verb names, and associate a numeric constant with each verb name. At least one verb MUST be specified, and a maximum of 16,777,215 (2^^24 - 1) verbs MAY be specified. This enumerated list SHOULD be densely numbered and (i.e. valued from '1' to 'N', where 'N' is the total number of verbs defined in the macro).

6.1.6.1. Mapping of the Verb Name Field

The <verbName> field is case-sensitive, and SHOULD be set to the most appropriate string name for each application verb. If a readable string is defined in an authoritative document, then that string SHOULD be used. If no such string exists, then an appropriate but arbitrary string should be selected for this value.

Verb names MUST be unique for a particular parent application. Note that the special 'connect(0)' verb is implicitly defined for each application protocol. It is possible for an explicit definition of this verb (e.g. 'connect(8)' for http) to exist for a protocol, as well as the implicit 'connect(0)' verb.

6.1.6.2. Mapping of the Verb Enum Field

The <verbEnum> field MUST be unique for all verbs associated with a particular parent application. This field MUST contain a value between '1' and '16,777,215' inclusive.

6.2. Protocol Directory Requirements

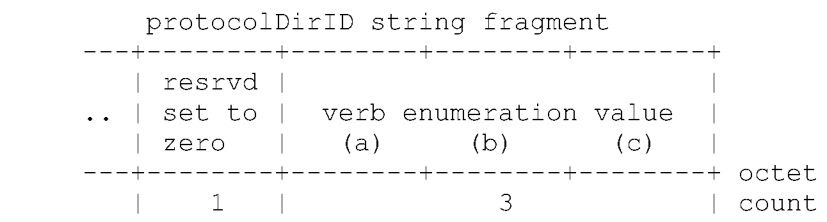
This section defines how the protocolDirTable should be populated for any application verb identified with a verb-identifier macro.

An agent MUST implement all applicable protocolDirTable MIB objects on behalf of each supported application verb.

6.2.1. Mapping of the Verb Layer Numbering Space

The verb layer consists of the 4 octets within the protocolDirID INDEX field which identify a particular application verb.

Figure 1
Verb Layer Format



The first octet is reserved for future use and MUST be set to zero.

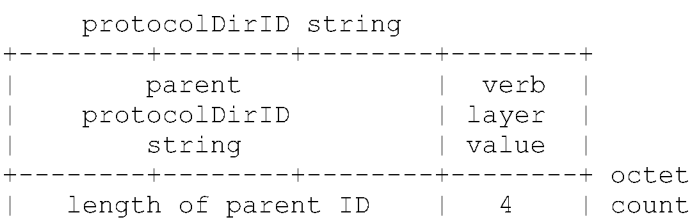
The next three octets identify the <verbEnum> field used to enumerate the particular application verb represented by the <verbName> field. This field is a 24-bit unsigned integer, encoded in network byte order.

The value zero is reserved to identify the special 'connect(0)' verb. This verb enumeration value (i.e. '0' part of 'connect(0)') MUST NOT be redefined in a verb identifier macro verb list. Note that the verb name 'connect' is not reserved and MAY be redefined in a verb list.

6.2.2. Mapping of the ProtocolDirID object

The protocolDirID OCTET STRING value for a particular application verb is represented by the protocolDirID value for the parent application, appended with the verb's layer identifier value.

Figure 2
ProtocolDirID Format for Verbs



The protocolDirID object is encoded as the protocolDirID value of the parent application, followed by four additional octets representing the verb layer. The verb layer value is encoded as [0.a.b.c] where 'a' is the high order byte, 'b' is the middle order byte, and 'c' is the low order byte of the <verbEnum> field for the specific application verb value. A valid PI verb enumeration will be encoded in the range "0.0.0.0" to "0.255.255.255", where the special value "0.0.0.0" is reserved for the implicitly defined 'connect(0)' verb.

6.2.3. Mapping of the ProtocolDirParameters object

The protocolDirParameters OCTET STRING value for a particular application verb is represented by the protocolDirParameters value for the parent application, appended with one octet containing the value zero.

6.2.4. Mapping of the ProtocolDirLocalIndex object

The agent MUST assign an appropriate protocolDirLocalIndex value for each application verb, according to the encoding rules defined for this object in [RFC2021] and [RFC2895].

6.2.5. Mapping of the protocolDirDescr object

The agent MUST convey the <verbName> value for a particular application verb in the protocolDirDescr object. This object SHOULD be encoded as

the protocolDirDescr value for the parent application, appended with a 'dot' character, followed by the exact text contained in the <verbName> field.

6.2.6. Mapping of the protocolDirType object

The agent MUST set the protocolDirType object for each application verb to the value representing the empty bit set ({}).

6.2.7. Mapping of the protocolDirAddressMapConfig object

The agent MUST set the protocolDirAddressMapConfig object for each application verb to the value 'notSupported(1)'.

6.2.8. Mapping of the protocolDirHostConfig object

The agent MUST set the protocolDirHostConfig object for each application verb present in the protocol directory, according to the monitoring capabilities for each verb. The agent MAY set this object to the same value as configured in the parent application protocolDirHostConfig object. The agent MAY choose to transition this object from the value 'supportedOn(2)' to 'supportedOff(3)', if the parent application protocolDirHostConfig object first transitions from 'supportedOn(2)' to 'supportedOff(3)'.

6.2.9. Mapping of the protocolDirMatrixConfig object

The agent MUST set the protocolDirMatrixConfig object for each application verb, according to the monitoring capabilities for each verb. The agent MAY set this object to the same value as configured in the parent application protocolDirMatrixConfig object. The agent MAY choose to transition this object from the value 'supportedOn(2)' to 'supportedOff(3)', if the parent application protocolDirMatrixConfig object first transitions from 'supportedOn(2)' to 'supportedOff(3)'.

6.2.10. Mapping of the protocolDirOwner object

This object is encoded exactly the same for application verbs as for other protocolDirTable entries, according to the rules specified in the RMON-2 MIB [RFC2021].

6.2.11. Mapping of the protocolDirStatus object

This object is encoded exactly the same for application verbs as for other protocolDirTable entries, according to the rules specified in

RMON-2 MIB [RFC2021].

7. Implementation Considerations

This section discusses the implementation implications for agents which support verbs in the protocol directory, and the RMON collections which utilize the protocol directory.

7.1. Stateful Protocol Decoding

Implementations of the RMON-2 MIB for AL and NL protocols typically require little if any state to be maintained by the probe. The probe can generally decide whether to count a packet and its octets on the packet's own merits, without referencing or updating any state information.

Implementations of the RMON-2 MIB at the verb layer will, for many protocols, need to maintain state information in order to correctly classify a packet as "belonging" to one verb or another. The examples below illustrate this point.

For SNMP over UDP, a Response-PDU for an SNMP Get-PDU can't be distinguished from a Response-PDU for a Getnext-PDU. A probe would need to maintain state information in order to correlate a Response-PDU from B to A with a previous request from A to B.

For application protocols carried over a stream-based transport such as TCP, the information required to identify an application verb can span several packets. A probe would need to follow the transport-layer flow in order to correctly parse the application-layer data.

7.2. Packet Capture

For packet capture based on verb-layer protocol directory filtering, the decision to include a packet in the capture buffer may need to be deferred until the packet can be conclusively attributed to a particular verb. A probe may need to pre-buffer packets while deciding to include or exclude them from capture based on other packets that have not yet arrived.

7.3. RMON-2 MIB Collections

Data collections such as the protocol distribution or AL Host Table require that each packet is counted only once, i.e. a given packet is fully classified as a single protocol encapsulation, which resolves to a

single leaf entry in the protocol directory. Also, octet counters related to protocol classification are incremented by the entire size of packet, not just the octets associated with a particular encapsulation layer.

It is possible that particular application protocols will allow multiple types of verbs to be present is a single packet. In this case, the agent must choose one verb type, and therefore one protocol directory entry, in order to properly count such a packet.

It is an implementation-specific matter as to which verb type an agent selects to identify a packet, in the event more than one verb type is present in that packet. Some possible choices include:

- the first verb type encountered in the packet
- the verb type with the most instances in the packet
- the verb type using the largest number of octets in the packet
- the most 'interesting' verb type in the packet (based on knowledge of that application protocol).

8. Appendix A: Usage Examples

The following examples are listed to demonstrate how RMON verb identifiers are declared.

[ed. the WG needs to decide if verb macros should be declared in a separate RFC, the way the PI macros are split out from the PI reference document.]

8.1. FTP Example

This example defines verb enumeration values for the File Transfer Protocol, as defined in RFC 959 and updated by RFC 2228 and RFC 2640. Note that verb name strings specified in the <verbName> field are not limited to 4 characters in length. In the FTP protocol, all the command names are 4 characters in length, and the verb name string should match the official command name as closely as possible.

```
ftp VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for FTP is derived from the list
       of commands defined for the File Transfer Protocol,
       which are identified by case-insensitive strings.
       The commands are simply listed in the order found
       in the FTP documentation."
    REFERENCE
      "File Transfer Protocol, RFC 959, Section 4.1;
       FTP Security Extensions, RFC 2228, Section 3;
       Internationalization of the File Transfer Protocol,
       RFC 2640, Section 4.1."
    ::= {
          user(1),      -- USER NAME
          pass(2),      -- PASSWORD
          acct(3),      -- ACCOUNT
          cwd(4),       -- CHANGE WORKING DIRECTORY
          cdup(5),      -- CHANGE TO PARENT DIRECTORY
          smnt(6),      -- STRUCTURE MOUNT
          rein(7),      -- REINITIALIZE
          quit(8),      -- LOGOUT
          port(9),      -- DATA PORT
          pasv(10),     -- PASSIVE
          type(11),     -- REPRESENTATION TYPE
          stru(12),     -- FILE STRUCTURE
          mode(13),     -- TRANSFER MODE
          retr(14),     -- RETRIEVE
```

```
        stor(15),    -- STORE
        stou(16),    -- STORE UNIQUE
        appe(17),    -- APPEND (with create)
        allo(18),    -- ALLOCATE
        rest(19),    -- RESTART
        rnfr(20),    -- RENAME FROM
        rnto(21),    -- RENAME TO
        abor(22),    -- ABORT
        dele(23),    -- DELETE
        rmd(24),     -- REMOVE DIRECTORY
        mkd(25),     -- MAKE DIRECTORY
        pwd(26),     -- PRINT WORKING DIRECTORY
        list(27),    -- LIST
        nlst(28),    -- NAME LIST
        site(29),    -- SITE PARAMETERS
        syst(30),    -- SYSTEM
        stat(31),    -- STATUS
        help(32),    -- HELP
        noop(33),    -- NOOP
        auth(34),    -- AUTHENTICATION/SECURITY MECHANISM
        adat(35),    -- AUTHENTICATION/SECURITY DATA
        pbsz(36),    -- PROTECTION BUFFER SIZE
        prot(37),    -- DATA CHANNEL PROTECTION LEVEL
        ccc(38),     -- CLEAR COMMAND CHANNEL
        mic(39),     -- INTEGRITY PROTECTED COMMAND
        conf(40),    -- CONFIDENTIALITY PROTECTED COMMAND
        enc(41),     -- PRIVACY PROTECTED COMMAND
        lang(42)     -- LANGUAGE
    }
```

8.2. POP3 Example

This example defines verb enumeration values for the Post Office Protocol, Version 3, as defined in RFC 1939 and updated by RFC 2449.

```
pop3 VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for POP3 is derived from the list
       of commands defined for the Post Office Protocol,
       which are identified by case-insensitive strings.
       The commands are simply listed in the order found
       in the POP3 command summary."
    REFERENCE
      "Post Office Protocol, Version 3, RFC 1939, Section 9;
```

```
       POP3 Extension Mechanism, RFC 2449, Section 5."
    ::= {
          user(1),
          pass(2),
          quit(3),
          stat(4),
          list(5),
          retr(6),
          dele(7),
          noop(8),
          rset(9),
          apop(10),
          top(11),
          uidl(12),
          capa(13)
    }
```

8.3. SNMP Example

This example defines verb enumeration values for the Simple Network Management Protocol, as defined in RFC 1905.

```
snmp VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for SNMP is derived from the list
       of PDU transaction types in the Protocol Operations
       document for SNMPv2.  Note that the 'Response'
       and 'Report' PDUs are not considered verbs, but are
       classified as belonging to the transaction type
       associated with the request PDU."
    REFERENCE
      "Protocol Operations for Version 2 of the
       Simple Network Management Protocol (SNMPv2),
       RFC 1905, Section 3."
    ::= {
          get(1),
          get-next(2),
          get-bulk(3),
          set(4),
          inform-request(5),
          trap(6)
    }
```

8.4. HTTP Example

This example defines verb enumeration values for the Hypertext Transfer Protocol, version 1.1, as defined in RFC 2616.

```
http VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for HTTP is derived from the list
       of methods defined for the Hypertext Transfer Protocol,
       which are identified by case-sensitive strings.
       The commands are simply listed in the order found
       in the HTTP/1.1 documentation.  Methods commonly used
       in HTTP/1.0 are a proper subset of those used in HTTP/1.1.
       Both versions of the protocol are in current use."
    REFERENCE
       "Hypertext Transfer Protocol -- HTTP/1.1, RFC 2616,
       Section 9; Hypertext Transfer Protocol -- HTTP/1.0, RFC
       1945, Section 8."
    ::= {
          options(1),
          get(2),
          head(3),
          post(4),
          put(5),
          delete(6),
          trace(7),
          connect(8)  -- reserved for future use by HTTP/1.1
    }
```

8.5. SMTP Example

This example defines verb enumeration values for the Simple Mail Transfer Protocol as defined in RFC 2821.

```
smtp VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for SMTP is derived from the set of
       commands defined for the protocol.  These commands are identified
       by case-insensitive strings.  Commands are listed in the
       order found in RFC 2821.  The special "xcmd" verb is defined
       here as a catch-all for private-use commands, which must
       start with the letter 'X'."
    REFERENCE
      "Simple Mail Transfer Protocol -- RFC 2821, sections 4.1.1
```

```
      and 4.1.5."
   ::= {
         ehlo(1),    -- Extended HELLO (4.1.1.1)
         helo(2),    -- HELLO (4.1.1.1)
         mail(3),    -- MAIL (4.1.1.2)
         rcpt(4),    -- RECIPIENT (4.1.1.3)
         data(5),    -- DATA (4.1.1.4)
         rset(6),    -- RESET (4.1.1.5)
         vrfy(7),    -- VERIFY (4.1.1.6)
         expn(8),    -- EXPAND (4.1.1.7)
         help(9),    -- HELP (4.1.1.8)
         noop(10),   -- NOOP (4.1.1.9)
         quit(11),   -- QUIT (4.1.1.10)
         xcmd(12)    -- Catch-all for private-use "X" commands (4.1.5)
   }
```

9. Intellectual Property

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

10. Acknowledgements

This memo is a product of the RMONMIB WG.

11. References

[RFC1155]
Rose, M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", RFC 1155, Performance Systems International, Hughes LAN Systems, May 1990.

[RFC1157]
Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", RFC 1157, SNMP Research, Performance Systems International, Performance Systems International, MIT Laboratory for Computer Science, May 1990.

[RFC1212]
Rose, M., and K. McCloghrie, "Concise MIB Definitions", RFC 1212, Performance Systems International, Hughes LAN Systems, March 1991.

[RFC1215]
M. Rose, "A Convention for Defining Traps for use with the SNMP", RFC 1215, Performance Systems International, March 1991.

[RFC1901]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Introduction to Community-based SNMPv2", RFC 1901, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC1905]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1905, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC1906]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC2021]
S. Waldbusser, "Remote Network Monitoring MIB (RMON-2)", RFC 2021, International Network Services, January 1997.

[RFC2026]
Bradner, S., "The Internet Standards Process -- Revision 3", RFC 2026, Harvard University, October, 1996.

[RFC2119]
S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels" RFC 2119, Harvard University, March 1997.

[RFC2570]
Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction to Version 3 of the Internet-standard Network Management Framework", RFC 2570, SNMP Research, Inc., TIS Labs at Network Associates, Inc., Ericsson, Cisco Systems, April 1999.

[RFC2571]
Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing SNMP Management Frameworks", RFC 2571, Cabletron Systems, Inc., BMC Software, Inc., IBM T. J. Watson Research, April 1999.

[RFC2572]
Case, J., Harrington D., Presuhn R., and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", RFC 2572, SNMP Research, Inc., Cabletron Systems, Inc., BMC Software, Inc., IBM T. J. Watson Research, April 1999.

[RFC2573]
Levi, D., Meyer, P., and B. Stewart, "SNMPv3 Applications", RFC 2573, SNMP Research, Inc., Secure Computing Corporation, Cisco Systems, April 1999.

[RFC2574]
Blumenthal, U., and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", RFC 2574, IBM T. J. Watson Research, April 1999.

[RFC2575]
Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", RFC 2575, IBM T. J. Watson Research, BMC Software, Inc., Cisco Systems, Inc., April 1999.

[RFC2578]
McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Structure of Management Information Version 2

(SMIv2)", RFC 2578, STD 58, Cisco Systems, SNMPinfo, TU Braunschweig, SNMP Research, First Virtual Holdings, International Network Services, April 1999.

[RFC2579]
McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Textual Conventions for SMIv2", RFC 2579, STD 58, Cisco Systems, SNMPinfo, TU Braunschweig, SNMP Research, First Virtual Holdings, International Network Services, April 1999.

[RFC2580]
McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Conformance Statements for SMIv2", RFC 2580, STD 58, Cisco Systems, SNMPinfo, TU Braunschweig, SNMP Research, First Virtual Holdings, International Network Services, April 1999.

[RFC2895]
Bierman, A., Bucci, C., and R. Iddon, "Remote Network Monitoring MIB Protocol Identifiers", RFC 2895, Cisco Systems, 3Com, Inc., August 2000.

12. Security Considerations

This memo defines the structure of a portion of the Remote Monitoring MIB framework, but does not define any MIB objects, protocol operations, or other mechanisms which can potentially introduce new security risks into a managed system.

13. Author's Address

Andy Bierman
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA USA 95134
Phone: +1 408-527-3711
Email: abierman@cisco.com

Chris Bucci
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA USA 95134
Phone: +1 408-527-5337
Email: cbucci@cisco.com

Russell Dietz
Apptitude, Inc.
6330 San Ignacio Avenue
San Jose, CA USA 95119
Phone: +1 408-574-2256
Email: rsdietz@apptitude.com

Albin Warth
NetScout Systems, Inc.
4 Technology Park Drive
Westford, MA 01886
Phone: (978) 614-4306
Email: albin@netscout.com

14. Full Copyright Statement

Exhibit B

Internet-Drafts

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are not an archival document series. These documents should not be cited or quoted in any formal document. Unrevised documents placed in the Internet-Drafts directories have a maximum life of six months. After that time, they must be updated, or they will be deleted. Details of the expiration procedure can be found in "Guidelines to Authors of Internet-Drafts." After a document becomes an RFC, it will be replaced in the Internet-Drafts Directories with an announcement to that effect.

To post an Internet-Draft to the Internet-Drafts Directory, use the Internet-Drafts Submission Tool. If you are unable to use the tool, then send your draft to internet-drafts@ietf.org.

Information on the status of all Internet-Drafts can be obtained via the Internet-Drafts Database Interface. You can use this tool to display Internet-Drafts by category, or to search for Internet-Drafts based on one or more search parameters. Also available are (a) an Index of Current Internet Drafts, which lists all active Internet-Drafts within each IETF working group (including title, author(s), initial submission date, and filename), and (b) an Index of Current Internet Draft Abstracts, which includes all of the above plus an abstract for each document.

Information on the status of Internet-Drafts under review by the IESG can be obtained via the IETF Draft Tracker (I-D Tracker).

Internet-Drafts can also be retrieved from mirror sites using ftp. If you wish to use rsync to check the contents of the Internet-Drafts directory, then click here.

Authors of Internet-Drafts should be familiar with the following documents:

Guidelines to Authors of Internet-Drafts

Note: The "Guidelines" document includes text that refers to RFC 3978 "IETF Rights in Contributions" and RFC 3979 "Intellectual Property Rights in IETF Technology."

I-D-Checklist

All Internet-Drafts that are submitted to the IESG for consideration as RFCs must conform to the requirements specified in the I-D Checklist.

IETF Secretariat - Questions on Internet-Drafts should be sent to the Internet-Drafts Administrator. Please send problem reports to ietf-web@ietf.org.

Exhibit C

B. Fenner (for the IESG)
AT&T Labs - Research
October 13, 2006

Guidelines to Authors of Internet-Drafts

Table of Contents

TOC

1. Submissions

ALL submissions to the Internet-Draft directories must be sent to internet-drafts@ietf.org.

The Internet-Draft may be sent as an attachment to the email, or the email may contain a URL which points to a copy of the Internet-Draft stored on a server.

DO NOT SEND ZIP files. We will not open them.

TOC

2. General Considerations

The Internet-Drafts directories are available to provide authors with the ability to distribute and solicit comments on documents they may eventually submit to the

IESG or RFC Editor for publication as an RFC. Internet-Drafts are not an archival document series. These documents should not be cited or quoted in any formal document. Unrevised documents placed in the Internet-Drafts directories have a maximum life of 185 days. After that time, they must be updated, or they will be deleted. See **RFC 2026** [RFC2026] for more information. In exceptional circumstances, an extension of this lifetime is possible; see **Section 8** below. After a document becomes an RFC, it will be replaced in the Internet-Drafts directories with an announcement to that effect.

Internet-Drafts are generally in the format of an RFC, although they may be rough drafts when the first version is submitted. This format is specified fully in "Instructions to RFC Authors" (see **the RFC Editor's Web pages** and **[I-D.rfc-editor-rfc2223bis]**). In brief, an Internet-Draft must be submitted in ASCII text, and should be limited to 72 characters per line and 58 lines per page, followed by a formfeed character. Overstriking to achieve bolding or underlining is not acceptable.

PostScript and/or PDF are acceptable, but only when submitted with a matching ASCII version (even if figures must be deleted). PostScript and PDF should be formatted for use on 8.5x11 inch paper. If A4 paper is used, an image area no greater than 254mm high should be used to avoid printing extra pages when printed on 8.5x11 paper.

There are differences between the RFC and Internet-Draft format. Internet-Drafts are NOT RFCs and are NOT a numbered document series. The words "INTERNET-DRAFT" should appear in the upper left hand corner of the first page. The document should NOT refer to itself as an RFC or a draft RFC.

The Internet-Draft should neither state nor imply that it has any standards status; to do so conflicts with the role of the RFC Editor and the IESG. The title of the document should not imply a status. Avoid the use of the terms Standard, Proposed, Draft, Experimental, Historic, Required, Recommended, Elective, or Restricted in the title of the Internet-Draft. Indicating what status the document is aimed for is OK, but should be done with the words "Intended status: <status>".

TOC

3. IPR-Related Notices Required in Internet-Drafts

All Internet-Drafts must have the following intellectual property rights (IPR) statement on the first page:

"By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79."

(See **RFC 3978** [RFC3978] section 5.1.)

All Internet-Drafts must include the following statements. They are normally placed

near the end of the document, to avoid intermixing boilerplate with document text.

"Copyright (C) The IETF Trust (YYYY).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights."

"This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE."

(See **RFC 3978** [RFC3978] sections 5.4 and 5.5, and **[I-D.ietf-ipr-ietf-trust-update]** sections 2.6 and 2.7.) "YYYY" in the copyright statement should be replaced with the current year.

Any submission which does not include these statements will be returned to the submitter. The IETF Secretariat will NOT add this text.

Note that although these statements appear to be written in English, they are actually written in lawyerese. Although it's awfully tempting to modify them, e.g., to remove "or she" in a document that has only male authors, please don't. It adds too much overhead to the Internet-Draft submission process to evaluate any given boilerplate variation to decide whether or not it changes the meaning.

TOC

4. Optional IPR-Related Notices that May Be Included in Internet-Drafts

If the submitter desires to eliminate the IETF's right to make modifications and derivative works of an Internet-Draft, one of the following two notices may be included after the IPR statement:

a. "This document may not be modified, and derivative works of it may not be created, except to publish it as an RFC and to translate it into languages other than English."

b. "This document may not be modified, and derivative works of it may not be created."

In the cases of MIB or PIB modules and in other cases where the Contribution includes material that is meant to be extracted in order to be used, the following should be appended to the above statements:

"other than to extract section XX as-is for separate use."

Statement (a) is used if the contributor intends for the Internet-Draft to be published as an RFC. Statement (b) is used along with the publication limitation statement below when the contributor does not intend for the Internet-Draft to be published as an RFC.

These notices may not be used with any standards-track document or with most working group documents, except as discussed in **RFC 3978** [RFC3978] section 7.3, since the IETF must retain change control over its documents and the ability to augment, clarify and enhance the original contribution in accordance with the IETF Standards Process.

Statement (a) may be appropriate when republishing standards produced by other (non-IETF) standards organizations, industry consortia or companies. These are typically published as Informational RFCs, and do not require that change control be ceded to the IETF. Basically, documents of this type convey information for the Internet community. (See **RFC 3978** [RFC3978] sections 5.2 and 7.3.)

If the Contributor only wants the contribution to be made available in an Internet-Draft (i.e., does not want the Internet-Draft to be published as an RFC) then the contributor may include the following publication limitation statement after Statement (b):

"This document may only be posted in an Internet-Draft."

This notice can be used on Internet-Drafts that are intended to provide background information to educate and to facilitate discussions within IETF working groups but are not intended to be published as an RFCs. (See **RFC 3978** [RFC3978] section 5.3.)

TOC

5. Internet-Draft Boilerplate

The following verbatim statement must follow the IPR statement and optional notices (if any):

"Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html"

Any submission which does not include these statements will be returned to the submitter. The IETF Secretariat will NOT add this text.

TOC

6. Formatting

Every Internet-Draft must have an Abstract section. The Abstract should provide a concise and comprehensive overview of the purpose and contents of the entire document. Its purpose is to give a technically knowledgeable reader a general overview of the function of the document, to decide whether reading it will be useful. In addition to its function in the document, the Abstract section is used as a summary in publication announcements and in **the online index of Internet-Drafts**.

Composing a useful Abstract is a non-trivial writing task. Often, a satisfactory abstract can be constructed in part from material from the Introduction section, but a good abstract will be shorter, less detailed, and broader in scope than the Introduction. Simply copying and pasting the first few paragraphs of the Introduction is tempting, but it generally results in an Abstract that is both incomplete and redundant.

An Abstract will typically be 5-10 lines, but an Abstract of more than 20 lines or less than 3 lines is generally not acceptable.

An Abstract should be complete in itself, so it should contain no citations unless they are completely defined within the Abstract. Abbreviations appearing in the Abstract should generally be expanded in parentheses. There is a small set of reasonable exceptions to this rule; for example, readers don't need to be reminded of what "IP" or "TCP" or "MIB" means. In the end, therefore, this is a judgment call, but please err on the side of explicitness.

In addition, the Internet-Draft should contain a section giving name and contact information (postal mail, voice/fax number and/or e-mail) for the authors.

All Internet-Drafts must contain the full filename (beginning with draft- and including the version number) in the text of the document. The filename information should, at a minimum, appear on the first page (possibly with the title). Internet-Draft filenames use lowercase characters ONLY. See **Section 7** for more information on filenames.

It is strongly recommended that the draft include a notice (with email address) of where comments should be sent. For example, "Comments are solicited and should be addressed to the working group's mailing list at ___@_____ and/or the author (s)."

A document expiration date should appear on the first and last page of the Internet-Draft. The expiration date is 185 days following the submission of the document as an Internet-Draft. Use of the phrase "expires in six months" or "expires in 185 days" is not acceptable.

Internet-Drafts must be in ASCII. No 8-bit characters are currently allowed. If you need to include code points, a suggestion might be to use the unicode convention: U+XXXX, where X is a hexadecimal digit.

If the Internet-Drafts is longer than about 15 pages, please include, on the second page, a table of contents to make the document easier to reference.

TOC

7. Naming and Submitting

Internet-Draft filenames have four parts, separated with hyphens and which may themselves contain hyphens:

1. All Internet-Draft filenames begin with "draft"
2. Document source:

   Working Group
   : The string "ietf-" followed by the working group acronym.

   Other
   : A string identifying an IETF-related body. The currently allowed list is
     - iab
     - iana
     - iaoc
     - iesg
     - ietf-edu
     - ietf-iesg (Historic)
     - ietf-proto
     - ietf-tools
     - ietf-trust
     - irtf
     - rfc-editor
     - rfced (Historic)

   Individual
   : A string related to the name of one of the authors in some way. There are no mechanical rules for this string but objectionable or misleading strings are subject to change or removal at the discretion of the Secretariat.

3. Document name. For non-working group documents that are targeted at a working group, this string often begins with the working group abbreviation. This document name is a word or two that reflect what the draft is about.
4. Two digit decimal version number, starting with 00.

Note that the limit on the total length of a filename excluding the version number is 50 characters, and the only characters allowed in a filename are lowercase letters, numerals and hyphens. Internet-Draft filenames are never reused; if a previous document has used your desired filename you must pick another.

Exhibit C

For those authors submitting updates to existing Internet-Drafts, the choice of the file name is easily determined (up the version by 1). For new documents, submit the document with a suggested filename following the above rules. Note that if the suggested filename is not acceptable for some reason (e.g., not getting working group chair approval for a working group document), the document will have to be resubmitted with the actual file name.

If the document is a new one (i.e., starting with revision -00.txt) and is submitted as a working group document, the IETF Secretariat needs permission from the chair(s) of the wg to publish it as a working group document. Working group chairs should submit this permission at (or close to) the time that the draft is submitted for posting. To expedite the process, authors are encouraged to send the document to "internet-drafts@ietf.org" and at the same time cc: to the chair(s) of the working group. If the document is accepted as a working group document, then it will have the draft-ietf-<working group acronym> file name and will be announced on the working group mailing list by the IETF Secretariat. If the document is not accepted as a working group document, it will be processed as an individual submission, where the filename will be draft-<yourname>, as described above.

NOTE: Revision numbers are based on the filename (as in first, second, or third version of this document). If there is a filename change, the version number starts over at -00. Put another way, the prior version number will NOT be incremented when an Internet-Draft filename has changed, e.g., from an individual to a working group document. ALL FILES BEGIN at -00.

Before each IETF meeting, a deadline is announced for submitting documents ahead of time to be published for the meeting. For new documents, the deadline is one week earlier than for new versions of old drafts. Care should be taken when submitting an Internet-Draft near the deadline. The Secretariat includes a "grace period" after the cutoff time; while the auto response message changes right at the cutoff time, submissions that are received by the Secretariat during the grace period will still be posted. This grace period is normally more than sufficient to ensure that documents submitted close to the deadline are received and accepted for posting. The Secretariat receives hundreds of Internet-Drafts immediately preceding an IETF meeting, and it can take several days to process and post them all. If an Internet-Draft that was submitted very close to the deadline has not been posted and announced within three days, then the submitter should send a message to ietf-action@ietf.org (using the suggested subject line "Status of I-D Submission: <filename>") and reference the auto-response acknowledgement for the document in the body of the message. The Secretariat will be happy to investigate the situation and post any valid submission that was not posted in the first round.

When you submit an Internet-Draft, you will receive an auto-response message from the Secretariat acknowledging that your Internet-Draft has been received. The subject line of the message will read: [Auto Response] <subject of your original message>. If you do not receive an acknowledgement within 2 hours after submitting your Internet-Draft, then please contact the Secretariat by sending a note to "ietf-action@ietf.org". The suggested subject line for this note is: "Status of I-D Submission: <filename>". If you need to track the status of your Internet-Draft at a later date, then please send a note to "ietf-action@ietf.org" (using the suggested subject line "Status of I-D Submission: <filename>") and include the

auto-response acknowledgement for your document in the body.

TOC

8. Expiring

An Internet-Draft will expire exactly 185 days from the date that it is posted on the IETF Web site (<**http://www.ietf.org/ID.html**>) unless it is replaced by an updated version (in which case the clock will start all over again for the new version, and the old version will be removed from the Internet-Draft directories), or unless it is under official review by the IESG (i.e., a request to publish it has been submitted). Specifically, when an Internet-Draft enters the "Publication Requested" state in the I-D Tracker, it will not be expired until its status is resolved (e.g., it is published as an RFC). I-D Tracker states not associated with a formal request to publish a document (e.g., "AD is Watching") will not prevent an Internet-Draft from expiring after 185 days.

Internet-Drafts will not expire during the period surrounding an IETF meeting when posting of updates to Internet-Drafts is suspended (i.e., between the cutoff date for submission of updated drafts, which is two weeks prior to an IETF meeting, and the date that posting of Internet-Drafts resumes). All Internet-Drafts scheduled to expire during this period will expire on the day that the Secretariat once again begins posting Internet-Drafts.

When an Internet-Draft expires, a "tombstone" file will be created that includes the filename and version number of the Internet-Draft that has expired. The filename of the tombstone file will be the same as that of the expired Internet-Draft with the version number increased by one. If a revised version of an expired Internet-Draft is submitted for posting, then the revised version will replace the tombstone file and must have the same version number as that previously assigned to the tombstone file. Tombstone files will never expire and will always be available for reference unless they are replaced by updated versions of the subject Internet-Drafts.

An expired Internet-Draft may be unexpired when necessary to further the IETF's work, including IETF liaison with other standards bodies. Such action will be taken by request of an IESG member, a working group chair of the Internet-Draft's working group, or one of the document authors. Such a request may be overridden; e.g., a working group chair of the Internet-Draft's working group will be notified if an author requests unexpiration and may request that the action not occur. This request should be sent to "internet-drafts@ietf.org" (using the suggested subject line "Resurrect I-D <filename>") and should be from an author, a working group chair or an IESG member.

The expiration date of an unexpired Internet-Draft may be extended with the same rules as unexpiration. This request should be sent to "internet-drafts@ietf.org" (using the suggested subject line "Extend expiration date for <filename>") and should be from an author, a working group chair or an IESG member.

TOC

9. Intellectual Property Rights

If you think that you, your company, or anyone else owns a patent or other IPR on the work described in the draft, you should read carefully **RFC 3979** [RFC3979]. The first notice required in Internet-Drafts, described in **Section 3** of this document, obligates you to send an IPR disclosure under certain circumstances. Before submitting the draft, it would be a good advice to talk to the working group chair or area directors about it.

TOC

10. Further Reading

The IETF process is described in **RFC 2026** [RFC2026]. The IETF rules concerning copyright are described in **RFC 3978** [RFC3978] and **[I-D.ietf-ipr-ietf-trust-update]**. The IETF rules on IPR are described in **RFC 3979** [RFC3979]. RFC 3978 and 3979 are updates to RFC 2026 and obsolete section 10 of that document. This document is for helping authors. If you need the definitive rules, read RFC 2026, RFC 3978 and RFC 3979.

More good references when submitting a document to the IESG for publication as an RFC are the web page on Submitting Internet-Drafts to the IESG (<**http://www.ietf.org/ID-Checklist.html**>), the RFC Editor's Web pages on how to publish an RFC (<**http://www.rfc-editor.org/howtopub.html**>), and the Instructions to RFC Authors (**[I-D.rfc-editor-rfc2223bis]**). Henrik Levkowetz has written an excellent tool that checks many of these requirements; it is available at <**http://tools.ietf.org/tools/idnits/**>.

There are several tools to help the process of writing an Internet-Draft in this format; the RFC Editor has collected several pointers on their web page (<**http://www.rfc-editor.org/formatting.html**>).

TOC

11. References

[I-D.ietf-ipr-ietf-trust-update]	Bradner, S., "**RFC 3978 Update to recognize the IETF Trust**," draft-ietf-ipr-ietf-trust-update-03 (work in progress), September 2006.
[I-D.rfc-editor-rfc2223bis]	Reynolds, J. and R. Braden, "**Instructions to Request for Comments (RFC) Authors**," draft-rfc-editor-rfc2223bis-08 (work in progress), July 2004.
[RFC2026]	**Bradner, S.**, "**The Internet Standards Process -- Revision 3**," BCP 9, RFC 2026, October 1996
[RFC3978]	Bradner, S., "**IETF Rights in Contributions**," BCP 78, RFC 3978, March 2005.
[RFC3979]	Bradner, S., "**Intellectual Property Rights in IETF Technology**," BCP 79, RFC 3979, March 200

TOC

Appendix A. Change History

Changes from March 25, 2005 version to October 13, 2006 version:

- Relax rules for "document source" part of filename in **Section 7**.
- Updated to reflect changes described in **[I-D.ietf-ipr-ietf-trust-update]**.
- Loosen the "near the end of the document" text for boilerplate in **Section 3**.
- Changed idnits reference to tools.ietf.org

Changes from Feb 8, 2005 version to March 25, 2005 version:

- Update all references from RFCs 3667/3668 to 3978/3979.
- Update IPR boilerplate with words from RFC3978. Add a note that it's not appropriate to change the boilerplate, even if it seems wrong.
- Make it clear in the IPR section that the author is required to disclose IPR under certain circumstances by the 3978/3979 boilerplate.
- Add "Any submission which does not include these statements will be returned to the submitter. The IETF Secretariat will NOT add this text." to the section on Internet-Draft boilerplate too.
- Spell out exactly how drafts are named.
- Remove the option of asking the secretariat for an Internet-Draft name.
- Add the option for an author to un-expire or extend the expiration date of an Internet-Draft.
- Treat Postscript and PDF the same.
- Say "254mm" instead of "10 inches" since we're talking about metric paper sizes.
- Fix minor typos and make some wording changes in the section on Abstracts, making the text closer to 2223bis.
- Include documentation on the I-D deadline and how to check on I-Ds submitted near the deadline.
- Add a pointer to the RFC-Editor's formatting web page.

TOC

Author's Address

Bill Fenner (for the IESG)
AT&T Labs - Research
1 River Oaks Place
San Jose, CA 95134-1918
USA
Phone: +1 408 493-8505
Email: fenner@research.att.com

Exhibit D

Internet Draft

Andy Bierman
Chris Bucci
Cisco Systems, Inc.
Russell Dietz
Apptitude, Inc.
Albin Warth
Netscout Systems, Inc.
14 July 2000

Remote Monitoring MIB Extensions for Identifying Application Protocol Verbs

<draft-ietf-rmonmib-appverbs-00.txt>

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026 [RFC2026].

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet- Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Distribution of this document is unlimited. Please send comments to the RMONMIB WG mailing list <rmonmib@ietf.org>.

Exhibit D

1. Copyright Notice

2. Abstract

This memo defines an experimental portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes the algorithms required to identify protocol operations (verbs) within the protocol encapsulations managed with the Remote Network Monitoring MIB Version 2 [RFC2021].

3. Table of Contents

4. The SNMP Network Management Framework

The SNMP Management Framework presently consists of five major components:

- An overall architecture, described in RFC 2571 [RFC2571].
- Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIv1 and described in RFC 1155 [RFC1155], RFC 1212 [RFC1212] and RFC 1215 [RFC1215]. The second version, called SMIv2, is described in RFC 2578 [RFC2578], RFC 2579 [RFC2579] and RFC 2580 [RFC2580].
- Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in RFC 1157 [RFC1157]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in RFC 1901 [RFC1901] and RFC 1906 [RFC1906]. The third version of the message protocol is called SNMPv3 and described in RFC 1906 [RFC1906], RFC 2572 [RFC2572] and RFC 2574 [RFC2574].
- Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in RFC 1157 [RFC1157]. A second set of protocol operations and associated PDU formats is described in RFC 1905 [RFC1905].

- A set of fundamental applications described in RFC 2573 [RFC2573] and the view-based access control mechanism described in RFC 2575 [RFC2575].

A more detailed introduction to the current SNMP Management Framework can be found in RFC 2570 [RFC2570].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the mechanisms defined in the SMI.

This memo does not specify a MIB module.

5. Overview

There is a need for a standardized way of identifying the protocol operations defined for particular application protocols. Different protocol operations can have very different performance characteristics, and it is desirable to collect certain metrics at this level of granularity. This memo defines extensions to the existing protocol identifier structure [RFC2074], and is intended to update, not obsolete, the existing protocol identifier encoding rules.

5.1. Protocol Identifier Framework

The RMON Protocol Identifier (PI) structure [RFC2074] allows for a variable number of layer identifiers. Each layer contributes 4 octets to the protocolDirID OCTET STRING and one octet to the protocolDirParameters OCTET STRING. These two MIB objects comprise the index into the protocolDirTable [RFC2021], and represent a globally unique identifier for a particular protocol encapsulation (or set of encapsulations if the wildcard base layer is used).

5.2. Protocol Identifier Extensions for Application Verbs

The existing RMON protocol identifier architecture requires that an application verb be represented by one additional protocol layer, appended to the protocol identifier for the parent application. Since some application verbs are defined as strings which can exceed 4 octets in length, an integer mapping must be provided for each string. This memo specifies how the verb layer is structured, as well as a verb identifier macro syntax for specification of verb name to integer mappings.

5.3. Terms

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119. [RFC2119]

This document uses some terms defined in the RMON Protocol Identifier Reference document [PIREF], and some new terms that need introduction here.

5.3.1. Application Verb

Also called simply 'verb'. Refers to one of potentially many protocol operations that are defined by a particular application protocol.

Note that an application verb is not equivalent to an application protocol sub-command or opcode within a packet containing a PDU for the application. An application verb is a transaction type, and may involve several PDU types within the application protocol (e.g., SNMP Get-PDU and Response-PDU). In some applications, a verb may encompass protocol operations pertaining to more than one protocol entry in the protocol directory (e.g., ftp and ftp-data).

5.3.2. Parent Application

One of potentially many protocol encapsulations which identifies a particular application protocol. This term refers generically to any or all such encapsulations for a given set of application verbs.

5.3.3. Verb Layer

The portion of the protocol identifier octet string which identifies the application verb.

5.3.4. Verb Set

The group of verbs enumerated for a particular application protocol. The list of verb strings within a particular verb-identifier macro invocation is also called the verb set for that verb identifier.

5.4. Relationship to the RMON-2 MIB

The RMON-2 MIB [RFC2021] contains the protocolDirTable MIB objects used to identify all protocol encapsulations that can be monitored by a particular RMON agent.

This memo describes how these MIB objects are mapped by an implementation, for entries which identify application verbs. This document does not define any new MIB objects to identify application verbs.

5.5. Relationship to the RMON MIB Protocol Identifier Reference

The RMON MIB Protocol Identifier Reference [PIREF] defines the RMON Protocol Identifier Macro Specification Language, as well as the encoding rules for the ProtocolDirID and protocolDirParameters OCTET STRINGs.

This memo defines extensions to the Protocol Identifier Reference document for the identification of application verb information. It does not obsolete any portion of the Protocol Identifier Reference document.

6. Verb Identifier Macro Format

The following example is meant to introduce the verb-identifier macro. This macro-like construct is used to represent protocol verbs for a specific parent application.

6.1. Lexical Conventions

The following keyword is added to the PI language:

```
VERB-IDENTIFIER
```

6.2. Extended Grammar for the PI Language

The following is the extended BNF notation for the grammar with starting symbol <piFile>, for representing verb identifier macros. Note that only the term <piFile> is actually modified from the definition in [PIREF]. The <piDefinition> syntax is not reproduced here, since this memo is intended to extend that definition, not replace it.

```
-- a file containing one or more Protocol Identifier (PI) definitions
<piFile> = [ <piDefinition> | <piVerbDefinition> ]...

-- a PI definition
<piVerbDefinition> =
  <parentProtoName> "VERB-IDENTIFIER"
        "DESCRIPTION" string
      [ "REFERENCE" string ]
```

```
            "::=" "{" <verbList> "}"

    -- a list of verb identifier string
    <verbList> = <verbId> [ <wspace> "," <wspace> <verbId> ]...

    -- a verb identifier string
    <verbId> = <verbName> [<wspace>] "(" [<wspace>]
              <verbEnum> [<wspace>] ")" [<wspace>]

    -- a verb name
    <verbName> = lcname

    -- a verb enumeration
    <verbEnum> = <posNum>

    -- a positive integer
    <posNum> = any integer value greater than zero and
               less than 16,777,216

    -- <piDefinition> syntax is defined in [PIREF]
    -- <wspace> syntax is defined in [PIREF]
    -- lcname syntax is defined in [PIREF]
```

6.3. Mapping of the Parent Protocol Name

The "parentProtoName" value, called the "parent protocol name" shall be an ASCII string consisting of one up to 64 characters. The encoding rules are exactly as specified in section 6.2.4 of [PIREF], for the mapping of the protocol name field. If a <protoName> and a <parentProtoName> field contain the same value, then they refer to the same protocol.

A protocol identifier macro SHOULD exist in the <piFile> for at least one encapsulation of the parent application protocol, if any verb identifier macros referencing that parent application are present in the <piFile>.

6.4. Mapping of the DESCRIPTION Clause

The DESCRIPTION clause provides a textual description of the protocol verb set identified by this macro. Notice that it SHOULD NOT contain details about items covered by the DECODING and REFERENCE clauses.

The DESCRIPTION clause MUST be present in all verb-identifier macro declarations.

6.5. Mapping of the REFERENCE Clause

If a publicly available reference document exists for this set of application protocol verbs, it SHOULD be listed here. Typically this will be a URL if possible; if not then it will be the name and address of the controlling body.

The REFERENCE clause is optional, but SHOULD be implemented if an authoritative reference exists which specifies the application protocol verbs defined in the <verbList> section of this macro.

6.6. Mapping of the Verb List Clause

The verb list clause MUST be present, and is used to identify a list of application verb names, and associate a numeric constant with each verb name. At least one verb MUST be specified, and a maximum of 16,777,215 (2^^24 - 1) verbs MAY be specified. This enumerated list SHOULD be densely numbered (i.e., valued from '1' to 'N', where 'N' is the total number of verbs defined in the macro).

6.6.1. Mapping of the Verb Name Field

The <verbName> field is case-sensitive, and SHOULD be set to the most appropriate string name for each application verb. If a readable string is defined in an authoritative document, then that exact string SHOULD be used. If no such string exists, then an appropriate but arbitrary string should be selected for this value.

Verb names MUST be unique for a particular parent application.

6.6.2. Mapping of the Verb Enum Field

The <verbEnum> field MUST be unique for all verbs associated with a particular parent application. This field MUST contain a value between '1' and '16,777,215' inclusive.

7. Verb Identifiers in the protocolDirTable

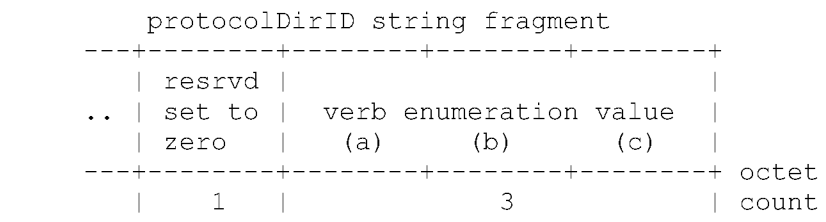
This section describes how the protocolDirTable should be populated for an application verb identified with a verb-identifier macro.

An agent MUST implement all applicable protocolDirTable MIB objects on behalf of each supported application verb.

7.1. Definition of the Verb Layer Numbering Space

The verb layer consists of the 4 octets within the protocolDirID INDEX field which identify a particular application verb.

Figure 1
Verb Layer Format



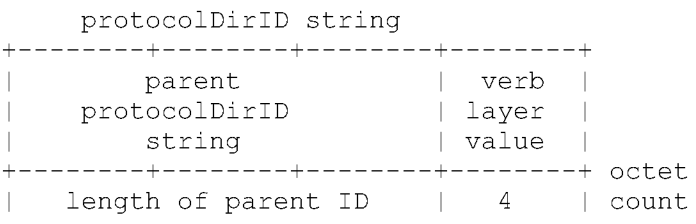
The first octet is a reserved field and MUST be set to zero.

The next three octets identify the <verbEnum> field used to enumerate the particular application verb represented by the <verbName> field. This field is a 24-bit unsigned integer, encoded in network byte order.

7.2. Mapping of the ProtocolDirID object

The protocolDirID OCTET STRING value for a particular application verb is represented by the protocolDirID value for the parent application, appended with the verb's layer identifier value.

Figure 2
ProtocolDirID Format for Verbs



The protocolDirID object is encoded as the protocolDirID value of the parent application, followed by four additional octets representing the

verb layer. The verb layer value is encoded as [0.a.b.c] where 'a' is the high order byte, 'b' is the middle order byte, and 'c' is the low order byte of the <verbEnum> field for the specific application verb value.

7.3. Mapping of the ProtocolDirParameters object

The protocolDirParameters OCTET STRING value for a particular application verb is represented by the protocolDirParameters value for the parent application, appended with one octet containing the value zero.

7.4. Mapping of the ProtocolDirLocalIndex object

The agent MUST assign an appropriate protocolDirLocalIndex value for each application verb, according to the encoding rules defined for this object in [RFC2021] and [PIREF].

7.5. Mapping of the protocolDirDescr object

The agent MUST convey the <verbName> value for a particular application verb in the protocolDirDescr object. This object SHOULD be encoded as the protocolDirDescr value for the parent application, appended with a 'dot' character, followed by the exact text contained in the <verbName> field.

7.6. Mapping of the protocolDirType object

The agent MUST set the protocolDirType object for each application verb to the value representing the empty bit set ({}).

7.7. Mapping of the protocolDirAddressMapConfig object

The agent MUST set the protocolDirAddressMapConfig object for each application verb to the value 'notSupported(1)'.

7.8. Mapping of the protocolDirHostConfig object

The agent MUST set the protocolDirHostConfig object for each application verb, according to the monitoring capabilities for the parent application. The agent SHOULD set this object to the same value as configured in the parent application protocolDirHostConfig object. The agent MAY choose to transition this object from the value 'supportedOn(2)' to 'supportedOff(3)', if the parent application protocolDirHostConfig object first transitions from 'supportedOn(2)' to

'supportedOff(3)'.

7.9. Mapping of the protocolDirMatrixConfig object

The agent MUST set the protocolDirMatrixConfig object for each application verb, according to the monitoring capabilities for the parent application. The agent SHOULD set this object to the same value as configured in the parent application protocolDirMatrixConfig object. The agent MAY choose to transition this object from the value 'supportedOn(2)' to 'supportedOff(3)', if the parent application protocolDirMatrixConfig object first transitions from 'supportedOn(2)' to 'supportedOff(3)'.

7.10. Mapping of the protocolDirOwner object

This object is encoded exactly the same for application verbs as for other protocolDirTable entries, according to the rules specified in the RMON-2 MIB [RFC2021].

7.11. Mapping of the protocolDirStatus object

This object is encoded exactly the same for application verbs as for other protocolDirTable entries, according to the rules specified in RMON-2 MIB [RFC2021].

8. Appendix A: Usage Examples

The following examples are listed to demonstrate how RMON verb identifiers are declared.

[ed. the WG needs to decide if verb macros should be declared in a separate RFC, the way the PI macros are split out from the PI reference document.]

8.1. FTP Example

This example defines verb enumeration values for the File Transfer Protocol, as defined in RFC 959 and updated by RFC 2228 and RFC 2640. Note that verb name strings specified in the <verbName> field are not limited to 4 characters in length. In the FTP protocol, all the command names are 4 characters in length, and the verb name string should match the official command name as closely as possible.

```
ftp VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for FTP is derived from the list
       of commands defined for the File Transfer Protocol,
       which are identified by case-insensitive strings.
       The commands are simply listed in the order found
       in the FTP documentation."
    REFERENCE
      "File Transfer Protocol, RFC 959, Section 4.1;
       FTP Security Extensions, RFC 2228, Section 3;
       Internationalization of the File Transfer Protocol,
       RFC 2640, Section 4.1."
    ::= {
          user(1),      -- USER NAME
          pass(2),      -- PASSWORD
          acct(3),      -- ACCOUNT
          cwd(4),       -- CHANGE WORKING DIRECTORY
          cdup(5),      -- CHANGE TO PARENT DIRECTORY
          smnt(6),      -- STRUCTURE MOUNT
          rein(7),      -- REINITIALIZE
          quit(8),      -- LOGOUT
          port(9),      -- DATA PORT
          pasv(10),     -- PASSIVE
          type(11),     -- REPRESENTATION TYPE
          stru(12),     -- FILE STRUCTURE
          mode(13),     -- TRANSFER MODE
          retr(14),     -- RETRIEVE
```

```
        stor(15),    -- STORE
        stou(16),    -- STORE UNIQUE
        appe(17),    -- APPEND (with create)
        allo(18),    -- ALLOCATE
        rest(19),    -- RESTART
        rnfr(20),    -- RENAME FROM
        rnto(21),    -- RENAME TO
        abor(22),    -- ABORT
        dele(23),    -- DELETE
        rmd(24),     -- REMOVE DIRECTORY
        mkd(25),     -- MAKE DIRECTORY
        pwd(26),     -- PRINT WORKING DIRECTORY
        list(27),    -- LIST
        nlst(28),    -- NAME LIST
        site(29),    -- SITE PARAMETERS
        syst(30),    -- SYSTEM
        stat(31),    -- STATUS
        help(32),    -- HELP
        noop(33),    -- NOOP
        auth(34),    -- AUTHENTICATION/SECURITY MECHANISM
        adat(35),    -- AUTHENTICATION/SECURITY DATA
        pbsz(36),    -- PROTECTION BUFFER SIZE
        prot(37),    -- DATA CHANNEL PROTECTION LEVEL
        ccc(38),     -- CLEAR COMMAND CHANNEL
        mic(39),     -- INTEGRITY PROTECTED COMMAND
        conf(40),    -- CONFIDENTIALITY PROTECTED COMMAND
        enc(41),     -- PRIVACY PROTECTED COMMAND
        lang(42)     -- LANGUAGE
    }
```

8.2. POP3 Example

This example defines verb enumeration values for the Post Office Protocol, Version 3, as defined in RFC 1939 and updated by RFC 2449.

```
pop3 VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for POP3 is derived from the list
       of commands defined for the Post Office Protocol,
       which are identified by case-insensitive strings.
       The commands are simply listed in the order found
       in the POP3 command summary."
    REFERENCE
      "Post Office Protocol, Version 3, RFC 1939, Section 9;
```

```
      POP3 Extension Mechanism, RFC 2449, Section 5."
    ::= {
          user(1),
          pass(2),
          quit(3),
          stat(4),
          list(5),
          retr(6),
          dele(7),
          noop(8),
          rset(9),
          apop(10),
          top(11),
          uidl(12),
          capa(13)
    }
```

8.3. SNMP Example

This example defines verb enumeration values for the Simple Network Management Protocol, as defined in RFC 1905.

```
snmp VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for SNMP is derived from the list
       of PDU transaction types in the Protocol Operations
       document for SNMPv2.  Note that the Response-PDU
       is not considered a verb, but is classified as
       belonging to the transaction type associated
       with the request PDU."
    REFERENCE
      "Protocol Operations for Version 2 of the
       Simple Network Management Protocol (SNMPv2),
       RFC 1905, Section 3."
    ::= {
          get(1),
          get-next(2),
          get-bulk(3),
          set(4),
          inform(5),
          trap(6),
          report(7)
    }
```

9. Intellectual Property

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat."

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

10. Acknowledgements

This memo is a product of the RMONMIB WG.

11. References

[PIREF]
Bierman, A., Bucci, C., and R. Iddon, "Remote Network Monitoring MIB Protocol Identifier Reference" <draft-ietf-rmonmib-rmonprot-ref-01.txt>, Cisco Systems, 3Com Corporation, June 1999.

[RFC1155]
Rose, M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", RFC 1155, Performance Systems International, Hughes LAN Systems, May 1990.

[RFC1157]
Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", RFC 1157, SNMP Research, Performance Systems International, Performance Systems International, MIT Laboratory for Computer Science, May 1990.

[RFC1212]
Rose, M., and K. McCloghrie, "Concise MIB Definitions", RFC 1212, Performance Systems International, Hughes LAN Systems, March 1991.

[RFC1215]
M. Rose, "A Convention for Defining Traps for use with the SNMP", RFC 1215, Performance Systems International, March 1991.

[RFC1901]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Introduction to Community-based SNMPv2", RFC 1901, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC1905]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1905, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC1906]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC2021]
S. Waldbusser, "Remote Network Monitoring MIB (RMON-2)", RFC 2021, International Network Services, January 1997.

[RFC2026]
Bradner, S., "The Internet Standards Process -- Revision 3", RFC 2026, Harvard University, October, 1996.

[RFC2074]
Bierman, A., and R. Iddon, "Remote Network Monitoring MIB Protocol Identifiers", RFC 2074, Cisco Systems, AXON Networks, Inc., January 1997.

[RFC2119]
S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels" RFC 2119, Harvard University, March 1997.

[RFC2570]
Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction to Version 3 of the Internet-standard Network Management Framework", RFC 2570, SNMP Research, Inc., TIS Labs at Network Associates, Inc., Ericsson, Cisco Systems, April 1999.

[RFC2571]
Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing SNMP Management Frameworks", RFC 2571, Cabletron Systems, Inc., BMC Software, Inc., IBM T. J. Watson Research, April 1999.

[RFC2572]
Case, J., Harrington D., Presuhn R., and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", RFC 2572, SNMP Research, Inc., Cabletron Systems, Inc., BMC Software, Inc., IBM T. J. Watson Research, April 1999.

[RFC2573]
Levi, D., Meyer, P., and B. Stewart, "SNMPv3 Applications", RFC 2573, SNMP Research, Inc., Secure Computing Corporation, Cisco Systems, April 1999.

[RFC2574]
Blumenthal, U., and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", RFC 2574, IBM T. J. Watson Research, April 1999.

[RFC2575]
Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", RFC 2575, IBM T. J. Watson Research, BMC Software, Inc., Cisco Systems, Inc., April 1999.

[RFC2578]
McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", RFC 2578, STD 58, Cisco Systems, SNMPinfo, TU Braunschweig, SNMP Research, First Virtual Holdings, International Network Services, April 1999.

[RFC2579]
McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Textual Conventions for SMIv2", RFC 2579, STD 58, Cisco Systems, SNMPinfo, TU Braunschweig, SNMP Research, First

Virtual Holdings, International Network Services, April 1999.

[RFC2580]
McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Conformance Statements for SMIv2", RFC 2580, STD 58, Cisco Systems, SNMPinfo, TU Braunschweig, SNMP Research, First Virtual Holdings, International Network Services, April 1999.

12. Security Considerations

This memo defines the structure of a portion of the Remote Monitoring MIB framework, but does not define any MIB objects, protocol operations, or other mechanisms which can potentially introduce new security risks into a managed system.

13. Author's Address

Andy Bierman
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA USA 95134
Phone: +1 408-527-3711
Email: abierman@cisco.com

Chris Bucci
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA USA 95134
Phone: +1 408-527-5337
Email: cbucci@cisco.com

Russell Dietz
Apptitude, Inc.
6330 San Ignacio Avenue
San Jose, CA USA 95119
Phone: +1 408-574-2256
Email: rsdietz@apptitude.com

Albin Warth
NetScout Systems, Inc.
4 Technology Park Drive
Westford, MA 01886
Phone: (978) 614-4306
Email: albin@netscout.com

Exhibit D

14. Full Copyright Statement

Exhibit E

Internet Draft

Andy Bierman
Chris Bucci
Cisco Systems, Inc.
Russell Dietz
Apptitude, Inc.
Albin Warth
Netscout Systems, Inc.
25 November 2000

Remote Monitoring MIB Extensions for Identifying Application Protocol Verbs

<draft-ietf-rmonmib-appverbs-01.txt>

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026 [RFC2026].

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet- Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Distribution of this document is unlimited. Please send comments to the RMONMIB WG mailing list <rmonmib@ietf.org>.

Exhibit E

1. Copyright Notice

2. Abstract

This memo defines an experimental portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes the algorithms required to identify protocol operations (verbs) within the protocol encapsulations managed with the Remote Network Monitoring MIB Version 2 [RFC2021].

3. Table of Contents

4. The SNMP Network Management Framework

The SNMP Management Framework presently consists of five major components:

- An overall architecture, described in RFC 2571 [RFC2571].
- Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIv1 and described in RFC 1155 [RFC1155], RFC 1212 [RFC1212] and RFC 1215 [RFC1215]. The second version, called SMIv2, is described in RFC 2578 [RFC2578], RFC 2579 [RFC2579] and RFC 2580 [RFC2580].
- Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in RFC 1157 [RFC1157]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in RFC 1901 [RFC1901] and RFC 1906 [RFC1906]. The third version of the message protocol is called SNMPv3 and described in RFC 1906 [RFC1906], RFC 2572 [RFC2572] and RFC 2574 [RFC2574].
- Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in RFC 1157 [RFC1157]. A second set of protocol operations and associated PDU formats is described in RFC 1905 [RFC1905].

- A set of fundamental applications described in RFC 2573 [RFC2573] and the view-based access control mechanism described in RFC 2575 [RFC2575].

A more detailed introduction to the current SNMP Management Framework can be found in RFC 2570 [RFC2570].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the mechanisms defined in the SMI.

This memo does not specify a MIB module.

5. Overview

There is a need for a standardized way of identifying the protocol operations defined for particular application protocols. Different protocol operations can have very different performance characteristics, and it is desirable to collect certain metrics at this level of granularity. This memo defines extensions to the existing protocol identifier structure [RFC2895], and is intended to update, not obsolete, the existing protocol identifier encoding rules.

5.1. Protocol Identifier Framework

The RMON Protocol Identifier (PI) structure [RFC2895] allows for a variable number of layer identifiers. Each layer contributes 4 octets to the protocolDirID OCTET STRING and one octet to the protocolDirParameters OCTET STRING. These two MIB objects comprise the index into the protocolDirTable [RFC2021], and represent a globally unique identifier for a particular protocol encapsulation (or set of encapsulations if the wild-card base layer is used).

5.2. Protocol Identifier Extensions for Application Verbs

The existing RMON protocol identifier architecture requires that an application verb be represented by one additional protocol layer, appended to the protocol identifier for the parent application. Since some application verbs are defined as strings which can exceed 4 octets in length, an integer mapping must be provided for each string. This memo specifies how the verb layer is structured, as well as a verb identifier macro syntax for specification of verb name to integer mappings.

5.3. Terms

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119. [RFC2119]

This document uses some terms defined in the RMON Protocol Identifier Reference document [RFC2895], and some new terms that need introduction here.

Application Verb
: Also called simply 'verb'. Refers to one of potentially many protocol operations that are defined by a particular application protocol.

  Note that an application verb is not equivalent to an application protocol sub-command or opcode within a packet containing a PDU for the application. An application verb is a transaction type, and may involve several PDU types within the application protocol (e.g., SNMP Get-PDU and Response-PDU). In some applications, a verb may encompass protocol operations pertaining to more than one protocol entry in the protocol directory (e.g., ftp and ftp-data).

Connect Verb
: The special application verb associated with connection or session setup and tear-down traffic, and not attributed to any other verb for the application. This verb is assigned the enumeration value of zero, and the verb 'connect(0)' is implicitly defined for all application protocols.

Parent Application
: One of potentially many protocol encapsulations which identifies a particular application protocol. This term refers generically to any or all such encapsulations for a given set of application verbs.

Verb Layer
: The portion of the protocol identifier octet string which identifies the application verb.

Verb Set
: The group of verbs enumerated for a particular application protocol. The list of verb strings within a particular verb-identifier macro invocation is also called the verb set for that verb identifier.

5.4. Relationship to the RMON-2 MIB

The RMON-2 MIB [RFC2021] contains the protocolDirTable MIB objects used to identify all protocol encapsulations that can be monitored by a particular RMON agent.

This memo describes how these MIB objects are mapped by an implementation, for entries which identify application verbs. This document does not define any new MIB objects to identify application verbs.

5.5. Relationship to the RMON MIB Protocol Identifier Reference

The RMON MIB Protocol Identifier Reference [RFC2895] defines the RMON Protocol Identifier Macro Specification Language, as well as the encoding rules for the ProtocolDirID and protocolDirParameters OCTET STRINGs.

This memo defines extensions to the Protocol Identifier Reference document for the identification of application verb information. It does not obsolete any portion of the Protocol Identifier Reference document.

6. Definitions

6.1. Verb Identifier Macro Format

The following example is meant to introduce the verb-identifier macro. This macro-like construct is used to represent protocol verbs for a specific parent application.

6.1.1. Lexical Conventions

The following keyword is added to the PI language:

```
VERB-IDENTIFIER
```

6.1.2. Extended Grammar for the PI Language

The following is the extended BNF notation for the grammar with starting symbol <piFile>, for representing verb identifier macros. Note that only the term <piFile> is actually modified from the definition in [RFC2895]. The <piDefinition> syntax is not reproduced here, since this memo is intended to extend that definition, not replace it.

```
    -- a file containing one or more
    -- Protocol Identifier (PI) definitions
    <piFile> = [ <piDefinition> | <piVerbDefinition> ]...

    -- a PI definition
    <piVerbDefinition> =
      <parentProtoName> "VERB-IDENTIFIER"
            "DESCRIPTION" string
          [ "REFERENCE" string ]
            "::=" "{" <verbList> "}"

    -- a list of verb identifier string
    <verbList> = <verbId> [ <wspace> "," <wspace> <verbId> ]...

    -- a verb identifier string
    <verbId> = <verbName> [<wspace>] "(" [<wspace>]
              <verbEnum> [<wspace>] ")" [<wspace>]

    -- a verb name
    <verbName> = lcname

    -- a verb enumeration
    <verbEnum> = <posNum>

    -- a positive integer
    <posNum> = any integer value greater than zero and
               less than 16,777,216

    -- <piDefinition> syntax is defined in [RFC2895]
    -- <wspace> syntax is defined in [RFC2895]
    -- lcname syntax is defined in [RFC2895]
```

6.1.3. Mapping of the Parent Protocol Name

The "parentProtoName" value, called the "parent protocol name" MUST be an ASCII string consisting of 1 to 64 characters. The encoding rules are exactly as specified in section 6.2.4 of [RFC2895], for the mapping of the protocol name field. If a <protoName> and a <parentProtoName> field contain the same value, then they refer to the same protocol.

A protocol identifier macro SHOULD exist in the <piFile> for at least one encapsulation of the parent application protocol, if any verb identifier macros referencing that parent application are present in the

<piFile>.

6.1.4. Mapping of the DESCRIPTION Clause

The DESCRIPTION clause provides a textual description of the protocol verb set identified by this macro. It SHOULD NOT contain details about items covered by the DECODING and REFERENCE clauses. The DESCRIPTION clause MUST be present in all verb-identifier macro declarations.

6.1.5. Mapping of the REFERENCE Clause

If a publicly available reference document exists for this set of application protocol verbs, it SHOULD be listed here. Typically this will be a URL, otherwise it will be the name and address of the controlling body.

The REFERENCE clause is optional, but SHOULD be implemented if an authoritative reference exists which specifies the application protocol verbs defined in the <verbList> section of this macro.

6.1.6. Mapping of the Verb List Clause

The verb list clause MUST be present, and is used to identify a list of application verb names, and associate a numeric constant with each verb name. At least one verb MUST be specified, and a maximum of 16,777,215 (2^^24 - 1) verbs MAY be specified. This enumerated list SHOULD be densely numbered and (i.e. valued from '1' to 'N', where 'N' is the total number of verbs defined in the macro).

6.1.6.1. Mapping of the Verb Name Field

The <verbName> field is case-sensitive, and SHOULD be set to the most appropriate string name for each application verb. If a readable string is defined in an authoritative document, then that string SHOULD be used. If no such string exists, then an appropriate but arbitrary string should be selected for this value.

Verb names MUST be unique for a particular parent application. Note that the special 'connect(0)' verb is implicitly defined for each application protocol. It is possible for an explicit definition of this verb (e.g. 'connect(8)' for http) to exist for a protocol, as well as the implicit 'connect(0)' verb.

6.1.6.2. Mapping of the Verb Enum Field

The <verbEnum> field MUST be unique for all verbs associated with a particular parent application. This field MUST contain a value between '1' and '16,777,215' inclusive.

6.2. Protocol Directory Requirements

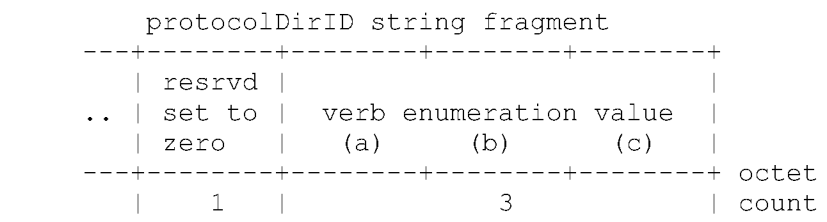
This section defines how the protocolDirTable should be populated for any application verb identified with a verb-identifier macro.

An agent MUST implement all applicable protocolDirTable MIB objects on behalf of each supported application verb.

6.2.1. Mapping of the Verb Layer Numbering Space

The verb layer consists of the 4 octets within the protocolDirID INDEX field which identify a particular application verb.

Figure 1
Verb Layer Format



The first octet is reserved for future use and MUST be set to zero.

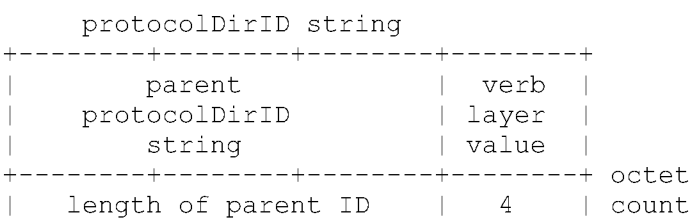
The next three octets identify the <verbEnum> field used to enumerate the particular application verb represented by the <verbName> field. This field is a 24-bit unsigned integer, encoded in network byte order.

The value zero is reserved to identify the special 'connect(0)' verb. This verb enumeration value (i.e. '0' part of 'connect(0)') MUST NOT be redefined in a verb identifier macro verb list. Note that the verb name 'connect' is not reserved and MAY be redefined in a verb list.

6.2.2. Mapping of the ProtocolDirID object

The protocolDirID OCTET STRING value for a particular application verb is represented by the protocolDirID value for the parent application, appended with the verb's layer identifier value.

Figure 2
ProtocolDirID Format for Verbs



The protocolDirID object is encoded as the protocolDirID value of the parent application, followed by four additional octets representing the verb layer. The verb layer value is encoded as [0.a.b.c] where 'a' is the high order byte, 'b' is the middle order byte, and 'c' is the low order byte of the <verbEnum> field for the specific application verb value. A valid PI verb enumeration will be encoded in the range "0.0.0.0" to "0.255.255.255", where the special value "0.0.0.0" is reserved for the implicitly defined 'connect(0)' verb.

6.2.3. Mapping of the ProtocolDirParameters object

The protocolDirParameters OCTET STRING value for a particular application verb is represented by the protocolDirParameters value for the parent application, appended with one octet containing the value zero.

6.2.4. Mapping of the ProtocolDirLocalIndex object

The agent MUST assign an appropriate protocolDirLocalIndex value for each application verb, according to the encoding rules defined for this object in [RFC2021] and [RFC2895].

6.2.5. Mapping of the protocolDirDescr object

The agent MUST convey the <verbName> value for a particular application verb in the protocolDirDescr object. This object SHOULD be encoded as

the protocolDirDescr value for the parent application, appended with a 'dot' character, followed by the exact text contained in the <verbName> field.

6.2.6. Mapping of the protocolDirType object

The agent MUST set the protocolDirType object for each application verb to the value representing the empty bit set ({}).

6.2.7. Mapping of the protocolDirAddressMapConfig object

The agent MUST set the protocolDirAddressMapConfig object for each application verb to the value 'notSupported(1)'.

6.2.8. Mapping of the protocolDirHostConfig object

The agent MUST set the protocolDirHostConfig object for each application verb present in the protocol directory, according to the monitoring capabilities for each verb. The agent MAY set this object to the same value as configured in the parent application protocolDirHostConfig object. The agent MAY choose to transition this object from the value 'supportedOn(2)' to 'supportedOff(3)', if the parent application protocolDirHostConfig object first transitions from 'supportedOn(2)' to 'supportedOff(3)'.

6.2.9. Mapping of the protocolDirMatrixConfig object

The agent MUST set the protocolDirMatrixConfig object for each application verb, according to the monitoring capabilities for each verb. The agent MAY set this object to the same value as configured in the parent application protocolDirMatrixConfig object. The agent MAY choose to transition this object from the value 'supportedOn(2)' to 'supportedOff(3)', if the parent application protocolDirMatrixConfig object first transitions from 'supportedOn(2)' to 'supportedOff(3)'.

6.2.10. Mapping of the protocolDirOwner object

This object is encoded exactly the same for application verbs as for other protocolDirTable entries, according to the rules specified in the RMON-2 MIB [RFC2021].

6.2.11. Mapping of the protocolDirStatus object

This object is encoded exactly the same for application verbs as for other protocolDirTable entries, according to the rules specified in

RMON-2 MIB [RFC2021].

7. Implementation Considerations

This section discusses the implementation implications for agents which support verbs in the protocol directory, and the RMON collections which utilize the protocol directory.

7.1. Stateful Protocol Decoding

Implementations of the RMON-2 MIB for AL and NL protocols typically require little if any state to be maintained by the probe. The probe can generally decide whether to count a packet and its octets on the packet's own merits, without referencing or updating any state information.

Implementations of the RMON-2 MIB at the verb layer will, for many protocols, need to maintain state information in order to correctly classify a packet as "belonging" to one verb or another. The examples below illustrate this point.

For SNMP over UDP, a Response-PDU for an SNMP Get-PDU can't be distinguished from a Response-PDU for a Getnext-PDU. A probe would need to maintain state information in order to correlate a Response-PDU from B to A with a previous request from A to B.

For application protocols carried over a stream-based transport such as TCP, the information required to identify an application verb can span several packets. A probe would need to follow the transport-layer flow in order to correctly parse the application-layer data.

7.2. Packet Capture

For packet capture based on verb-layer protocol directory filtering, the decision to include a packet in the capture buffer may need to be deferred until the packet can be conclusively attributed to a particular verb. A probe may need to pre-buffer packets while deciding to include or exclude them from capture based on other packets that have not yet arrived.

7.3. RMON-2 MIB Collections

Data collections such as the protocol distribution or AL Host Table require that each packet is counted only once, i.e. a given packet is fully classified as a single protocol encapsulation, which resolves to a

single leaf entry in the protocol directory. Also, octet counters related to protocol classification are incremented by the entire size of packet, not just the octets associated with a particular encapsulation layer.

It is possible that particular application protocols will allow multiple types of verbs to be present is a single packet. In this case, the agent must choose one verb type, and therefore one protocol directory entry, in order to properly count such a packet.

It is an implementation-specific matter as to which verb type an agent selects to identify a packet, in the event more than one verb type is present in that packet. Some possible choices include:

- the first verb type encountered in the packet
- the verb type with the most instances in the packet
- the verb type using the largest number of octets in the packet
- the most 'interesting' verb type in the packet (based on knowledge of that application protocol).

8. Appendix A: Usage Examples

The following examples are listed to demonstrate how RMON verb identifiers are declared.

[ed. the WG needs to decide if verb macros should be declared in a separate RFC, the way the PI macros are split out from the PI reference document.]

8.1. FTP Example

This example defines verb enumeration values for the File Transfer Protocol, as defined in RFC 959 and updated by RFC 2228 and RFC 2640. Note that verb name strings specified in the <verbName> field are not limited to 4 characters in length. In the FTP protocol, all the command names are 4 characters in length, and the verb name string should match the official command name as closely as possible.

```
ftp VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for FTP is derived from the list
       of commands defined for the File Transfer Protocol,
       which are identified by case-insensitive strings.
       The commands are simply listed in the order found
       in the FTP documentation."
    REFERENCE
      "File Transfer Protocol, RFC 959, Section 4.1;
       FTP Security Extensions, RFC 2228, Section 3;
       Internationalization of the File Transfer Protocol,
       RFC 2640, Section 4.1."
    ::= {
          user(1),      -- USER NAME
          pass(2),      -- PASSWORD
          acct(3),      -- ACCOUNT
          cwd(4),       -- CHANGE WORKING DIRECTORY
          cdup(5),      -- CHANGE TO PARENT DIRECTORY
          smnt(6),      -- STRUCTURE MOUNT
          rein(7),      -- REINITIALIZE
          quit(8),      -- LOGOUT
          port(9),      -- DATA PORT
          pasv(10),     -- PASSIVE
          type(11),     -- REPRESENTATION TYPE
          stru(12),     -- FILE STRUCTURE
          mode(13),     -- TRANSFER MODE
          retr(14),     -- RETRIEVE
```

```
        stor(15),    -- STORE
        stou(16),    -- STORE UNIQUE
        appe(17),    -- APPEND (with create)
        allo(18),    -- ALLOCATE
        rest(19),    -- RESTART
        rnfr(20),    -- RENAME FROM
        rnto(21),    -- RENAME TO
        abor(22),    -- ABORT
        dele(23),    -- DELETE
        rmd(24),     -- REMOVE DIRECTORY
        mkd(25),     -- MAKE DIRECTORY
        pwd(26),     -- PRINT WORKING DIRECTORY
        list(27),    -- LIST
        nlst(28),    -- NAME LIST
        site(29),    -- SITE PARAMETERS
        syst(30),    -- SYSTEM
        stat(31),    -- STATUS
        help(32),    -- HELP
        noop(33),    -- NOOP
        auth(34),    -- AUTHENTICATION/SECURITY MECHANISM
        adat(35),    -- AUTHENTICATION/SECURITY DATA
        pbsz(36),    -- PROTECTION BUFFER SIZE
        prot(37),    -- DATA CHANNEL PROTECTION LEVEL
        ccc(38),     -- CLEAR COMMAND CHANNEL
        mic(39),     -- INTEGRITY PROTECTED COMMAND
        conf(40),    -- CONFIDENTIALITY PROTECTED COMMAND
        enc(41),     -- PRIVACY PROTECTED COMMAND
        lang(42)     -- LANGUAGE
    }
```

8.2. POP3 Example

This example defines verb enumeration values for the Post Office Protocol, Version 3, as defined in RFC 1939 and updated by RFC 2449.

```
pop3 VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for POP3 is derived from the list
       of commands defined for the Post Office Protocol,
       which are identified by case-insensitive strings.
       The commands are simply listed in the order found
       in the POP3 command summary."
    REFERENCE
      "Post Office Protocol, Version 3, RFC 1939, Section 9;
```

```
       POP3 Extension Mechanism, RFC 2449, Section 5."
    ::= {
          user(1),
          pass(2),
          quit(3),
          stat(4),
          list(5),
          retr(6),
          dele(7),
          noop(8),
          rset(9),
          apop(10),
          top(11),
          uidl(12),
          capa(13)
    }
```

8.3. SNMP Example

This example defines verb enumeration values for the Simple Network Management Protocol, as defined in RFC 1905.

```
snmp VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for SNMP is derived from the list
       of PDU transaction types in the Protocol Operations
       document for SNMPv2.  Note that the 'Response'
       and 'Report' PDUs are not considered verbs, but are
       classified as belonging to the transaction type
       associated with the request PDU."
    REFERENCE
      "Protocol Operations for Version 2 of the
       Simple Network Management Protocol (SNMPv2),
       RFC 1905, Section 3."
    ::= {
          get(1),
          get-next(2),
          get-bulk(3),
          set(4),
          inform-request(5),
          trap(6)
    }
```

8.4. HTTP Example

This example defines verb enumeration values for the Hypertext Transfer Protocol, version 1.1, as defined in RFC 2616.

```
http VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for HTTP is derived from the list
       of methods defined for the Hypertext Transfer Protocol,
       which are identified by case-sensitive strings.
       The commands are simply listed in the order found
       in the HTTP/1.1 documentation.  Methods commonly used
       in HTTP/1.0 are a proper subset of those used in HTTP/1.1.
       Both versions of the protocol are in current use."
    REFERENCE
       "Hypertext Transfer Protocol -- HTTP/1.1, RFC 2616,
       Section 9; Hypertext Transfer Protocol -- HTTP/1.0, RFC
       1945, Section 8."
    ::= {
          options(1),
          get(2),
          head(3),
          post(4),
          put(5),
          delete(6),
          trace(7),
          connect(8)  -- reserved for future use by HTTP/1.1
    }
```

9. Intellectual Property

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

10. Acknowledgements

This memo is a product of the RMONMIB WG.

11. References

[RFC1155]
Rose, M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", RFC 1155, Performance Systems International, Hughes LAN Systems, May 1990.

[RFC1157]
Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", RFC 1157, SNMP Research, Performance Systems International, Performance Systems International, MIT Laboratory for Computer Science, May 1990.

[RFC1212]
Rose, M., and K. McCloghrie, "Concise MIB Definitions", RFC 1212, Performance Systems International, Hughes LAN Systems, March 1991.

[RFC1215]
M. Rose, "A Convention for Defining Traps for use with the SNMP", RFC 1215, Performance Systems International, March 1991.

[RFC1901]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Introduction to Community-based SNMPv2", RFC 1901, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC1905]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1905, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

Exhibit E

[RFC1906]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC2021]
S. Waldbusser, "Remote Network Monitoring MIB (RMON-2)", RFC 2021, International Network Services, January 1997.

[RFC2026]
Bradner, S., "The Internet Standards Process -- Revision 3", RFC 2026, Harvard University, October, 1996.

[RFC2119]
S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels" RFC 2119, Harvard University, March 1997.

[RFC2570]
Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction to Version 3 of the Internet-standard Network Management Framework", RFC 2570, SNMP Research, Inc., TIS Labs at Network Associates, Inc., Ericsson, Cisco Systems, April 1999.

[RFC2571]
Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing SNMP Management Frameworks", RFC 2571, Cabletron Systems, Inc., BMC Software, Inc., IBM T. J. Watson Research, April 1999.

[RFC2572]
Case, J., Harrington D., Presuhn R., and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", RFC 2572, SNMP Research, Inc., Cabletron Systems, Inc., BMC Software, Inc., IBM T. J. Watson Research, April 1999.

[RFC2573]
Levi, D., Meyer, P., and B. Stewart, "SNMPv3 Applications", RFC 2573, SNMP Research, Inc., Secure Computing Corporation, Cisco Systems, April 1999.

[RFC2574]
Blumenthal, U., and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", RFC

2574, IBM T. J. Watson Research, April 1999.

[RFC2575]
Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", RFC 2575, IBM T. J. Watson Research, BMC Software, Inc., Cisco Systems, Inc., April 1999.

[RFC2578]
McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", RFC 2578, STD 58, Cisco Systems, SNMPinfo, TU Braunschweig, SNMP Research, First Virtual Holdings, International Network Services, April 1999.

[RFC2579]
McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Textual Conventions for SMIv2", RFC 2579, STD 58, Cisco Systems, SNMPinfo, TU Braunschweig, SNMP Research, First Virtual Holdings, International Network Services, April 1999.

[RFC2580]
McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Conformance Statements for SMIv2", RFC 2580, STD 58, Cisco Systems, SNMPinfo, TU Braunschweig, SNMP Research, First Virtual Holdings, International Network Services, April 1999.

[RFC2895]
Bierman, A., Bucci, C., and R. Iddon, "Remote Network Monitoring MIB Protocol Identifiers", RFC 2895, Cisco Systems, 3Com, Inc., August 2000.

12. Security Considerations

This memo defines the structure of a portion of the Remote Monitoring MIB framework, but does not define any MIB objects, protocol operations, or other mechanisms which can potentially introduce new security risks into a managed system.

13. Author's Address

Andy Bierman
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA USA 95134
Phone: +1 408-527-3711
Email: abierman@cisco.com

Chris Bucci
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA USA 95134
Phone: +1 408-527-5337
Email: cbucci@cisco.com

Russell Dietz
Apptitude, Inc.
6330 San Ignacio Avenue
San Jose, CA USA 95119
Phone: +1 408-574-2256
Email: rsdietz@apptitude.com

Albin Warth
NetScout Systems, Inc.
4 Technology Park Drive
Westford, MA 01886
Phone: (978) 614-4306
Email: albin@netscout.com

14. Full Copyright Statement

Exhibit F

Network Working Group
Request for Comments: 3395
Updates: 2895
Category: Standards Track

A. Bierman
C. Bucci
Cisco Systems, Inc.
R. Dietz
Hifn, Inc.
A. Warth

September 2002

Remote Network Monitoring MIB Protocol Identifier Reference Extensions

Status of this Memo

Copyright Notice

Abstract

This memo defines extensions to the Protocol Identifier Reference document for the identification of application verb information. It updates the Protocol Identifier Reference document but does not obsolete any portion of that document. In particular, it describes the algorithms required to identify protocol operations (verbs) within the protocol encapsulations managed with MIBs such as the Remote Network Monitoring MIB Version 2, RFC 2021.

Table of Contents

Exhibit F

1. The SNMP Network Management Framework

The SNMP Management Framework presently consists of five major components:

- An overall architecture, described in RFC 2571 [RFC2571].
- Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIv1 and is described in STD 16, RFC 1155 [RFC1155], STD 16, RFC 1212 [RFC1212] and

RFC 1215 [RFC1215]. The second version, called SMIv2, is described in STD 58, RFC 2578 [RFC2578], RFC 2579 [RFC2579] and RFC 2580 [RFC2580].

- Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and is described in STD 15, RFC 1157 [RFC1157]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and is described in RFC 1901 [RFC1901] and RFC 1906 [RFC1906]. The third version of the message protocol is called SNMPv3 and is described in RFC 1906 [RFC1906], RFC 2572 [RFC2572] and RFC 2574 [RFC2574].

- Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in STD 15, RFC 1157 [RFC1157]. A second set of protocol operations and associated PDU formats is described in RFC 1905 [RFC1905].

- A set of fundamental applications is described in RFC 2573 [RFC2573]. The view-based access control mechanism is described in RFC 2575 [RFC2575].

A more detailed introduction to the current SNMP Management Framework can be found in RFC 2570 [RFC2570].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the mechanisms defined in the SMI.

This memo does not specify a MIB module.

2. Overview

There is a need for a standardized way of identifying the protocol operations defined for particular application protocols. Different protocol operations can have very different performance characteristics, and it is desirable to collect certain metrics at this level of granularity. This memo defines extensions to the existing protocol identifier structure [RFC2895] and is intended to update, not obsolete, the existing protocol identifier encoding rules.

2.1 Protocol Identifier Framework

The RMON Protocol Identifier (PI) structure [RFC2895] allows for a variable number of layer identifiers. Each layer contributes 4 octets to the protocolDirID OCTET STRING and one octet to the

protocolDirParameters OCTET STRING. These two MIB objects comprise the index in the protocolDirTable [RFC2021] and represent a globally unique identifier for a particular protocol encapsulation (or set of encapsulations if the wild-card base layer is used).

2.2 Protocol Identifier Extensions for Application Verbs

The existing RMON protocol identifier architecture requires that an application verb be represented by one additional protocol layer, appended to the protocol identifier for the parent application. Since some application verbs are defined as strings which can exceed 4 octets in length, an integer mapping must be provided for each string. This memo specifies how the verb layer is structured, as well as a verb identifier macro syntax for specification of verb name to integer mappings.

2.3 Terms

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

This document uses some terms defined in the RMON Protocol Identifier Reference document [RFC2895] and some new terms that need introduction here.

Application Verb
: Also called simply 'verb'. Refers to one of potentially many protocol operations that are defined by a particular application protocol.

  Note that an application verb is not equivalent to an application protocol sub-command or opcode within a packet containing a PDU for the application. An application verb is a transaction type and may involve several PDU types within the application protocol (e.g., SNMP Get-PDU and Response-PDU). In some applications, a verb may encompass protocol operations pertaining to more than one protocol entry in the protocol directory (e.g., ftp and ftp-data).

Connect Verb
: The special application verb associated with connection or session setup and tear-down traffic, and not attributed to any other verb for the application. This verb is assigned the enumeration value of zero, and the verb 'connect(0)' is implicitly defined for all application protocols.

Parent Application
One of potentially many protocol encapsulations which identifies a particular application protocol. This term refers generically to any or all such encapsulations for a given set of application verbs.

Verb Layer
The portion of the protocol identifier octet string which identifies the application verb.

Verb Set
The group of verbs enumerated for a particular application protocol. The list of verb strings within a particular verb-identifier macro invocation is also called the verb set for that verb identifier.

2.4 Relationship to the RMON-2 MIB

The RMON-2 MIB [RFC2021] contains the protocolDirTable MIB objects used to identify all protocol encapsulations that can be monitored by a particular RMON agent.

This memo describes how these MIB objects are mapped by an implementation for entries which identify application verbs. This document does not define any new MIB objects to identify application verbs. The applicability of the definitions in this document is not limited to the RMON-2 MIB. Other specifications which utilize the RMON-2 protocolDirTable and/or the protocol identifier macros which it represents can also utilize the application verb macro definitions contained in this document.

2.5 Relationship to the RMON MIB Protocol Identifier Reference

The RMON MIB Protocol Identifier Reference [RFC2895] defines the RMON Protocol Identifier Macro Specification Language as well as the encoding rules for the ProtocolDirID and protocolDirParameters OCTET STRINGs. This memo defines extensions to the Protocol Identifier Reference for the identification of application verb information. It does not obsolete any portion of the Protocol Identifier Reference document.

3. Definitions

3.1 Verb Identifier Macro Format

The following example is meant to introduce the verb-identifier macro. This macro-like construct is used to represent protocol verbs for a specific parent application.

3.1.1 Lexical Conventions

The following keyword is added to the PI language:

VERB-IDENTIFIER

3.1.2 Extended Grammar for the PI Language

The following is the extended BNF notation for the grammar with starting symbol <piFile>. It is for representing verb identifier macros. Note that only the term <piFile> is actually modified from the definition in [RFC2895]. The <piDefinition> syntax is not reproduced here, since this memo is intended to extend that definition, not replace it.

```
-- a file containing one or more
-- Protocol Identifier (PI) definitions
<piFile> = [ <piDefinition> | <piVerbDefinition> ]...

-- a PI definition
<piVerbDefinition> =
  [<wspace>] <parentProtoName> <wspace> "VERB-IDENTIFIER"
        <wspace> "DESCRIPTION" <wspace> string
      [ <wspace> "REFERENCE" <wspace> string ]
      [<wspace>] "::=" [<wspace>]
      "{" [<wspace>] <verbList> [<wspace>] "}" [<wspace>]

-- a list of verb identifier string
<verbList> = <verbId> [ [<wspace>] "," [<wspace>] <verbId> ]...

-- a verb identifier string
<verbId> = <verbName> [<wspace>] "(" [<wspace>]
          <verbEnum> [<wspace>] ")" [<wspace>]

-- a protocol name
<parentProtoName> = <protoName>

-- a verb name
<verbName> = <lcname>

-- a verb enumeration
<verbEnum> = <posNum>

-- a positive integer
<posNum> = any integer value greater than zero and
           less than 16,777,216

-- <piDefinition> syntax is defined in [RFC2895]
```

```
       -- <protoName> syntax is defined in [RFC2895]
       -- <wspace> syntax is defined in [RFC2895]
       -- <lcname> syntax is defined in [RFC2895]
```

3.1.3 Mapping of the Parent Protocol Name

The "parentProtoName" value, called the "parent protocol name", SHOULD be an ASCII string consisting of 1 to 64 characters. (These names are intended to appear in IETF documentation, so the use of UTF-8 is not appropriate.) The encoding rules are exactly as specified in section 6.2.4 of [RFC2895] for the mapping of the protocol name field. The value for <parentProtoName> (which is called the "parent protocol name") MUST be the value of a protocol identifier defined as specified for <protoName> in section 3.2.4 of [RFC2895]. The value of <parentProtoName> MUST specify a <protoName> defined in the <piFile>.

A protocol identifier macro SHOULD exist in the <piFile> for at least one encapsulation of the parent application protocol if any verb identifier macros referencing that parent application are present in the <piFile>.

3.1.4 Mapping of the DESCRIPTION Clause

The DESCRIPTION clause provides a textual description of the protocol verb set identified by this macro. It SHOULD NOT contain details about items covered by the REFERENCE clause. The DESCRIPTION clause MUST be present in all verb-identifier macro declarations.

3.1.5 Mapping of the REFERENCE Clause

If a publicly available reference document exists for this set of application protocol verbs, it SHOULD be listed here. Typically this will be a URL, otherwise it will be the name and address of the controlling body.

The REFERENCE clause is optional but SHOULD be present if an authoritative reference exists which specifies the application protocol verbs defined in the <verbList> section of this macro.

3.1.6 Mapping of the Verb List Clause

The verb list clause MUST be present. It is used to identify a list of application verb names and associate a numeric constant with each verb name. At least one verb MUST be specified and a maximum of 16,777,215 (2^^24 - 1) verbs MAY be specified. This enumerated list SHOULD be densely numbered (i.e., valued from '1' to 'N', where 'N' is the total number of verbs defined in the macro).

3.1.6.1 Mapping of the Verb Name Field

The <verbName> field is case-sensitive and SHOULD be set to the most appropriate string name for each application verb. If such a descriptive string is defined in an authoritative document then that string SHOULD be used. If no such string exists then an appropriate but arbitrary string should be selected for this value.

Verb names MUST be unique for a particular parent application. Note that the special 'connect(0)' verb is implicitly defined for each application protocol. It is possible for an explicit definition of this verb (e.g., 'connect(8)' for http) to exist for a protocol, as well as the implicit 'connect(0)' verb.

3.1.6.2 Mapping of the Verb Enum Field

The <verbEnum> field MUST be unique for all verbs associated with a particular parent application. This field SHOULD contain a value between '1' and '16,777,215' inclusive.

3.2 Protocol Directory Requirements

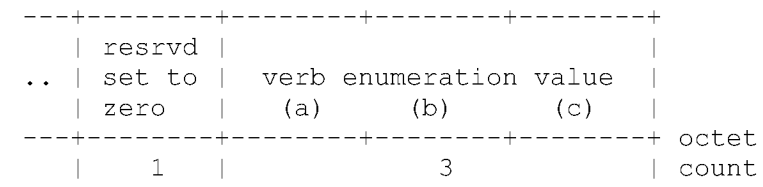
This section defines how the protocolDirTable should be populated for any application verb identified with a verb-identifier macro.

An agent MUST implement all applicable protocolDirTable MIB objects on behalf of each supported application verb.

3.2.1 Mapping of the Verb Layer Numbering Space

The verb layer consists of the 4 octets within the protocolDirID INDEX field which identify a particular application verb.

Figure 1
Verb Layer Format



The first octet is reserved for future use and MUST be set to zero.

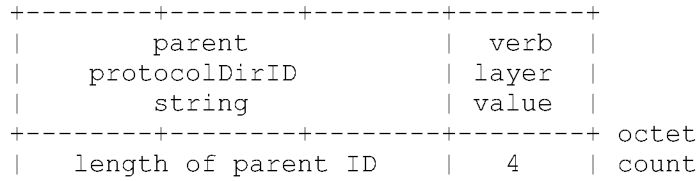
The next three octets identify the <verbEnum> field used to enumerate the particular application verb represented by the <verbName> field. This field is a 24-bit unsigned integer, encoded in network byte order.

The value zero is reserved to identify the special 'connect(0)' verb. This verb enumeration value (i.e., '0' part of 'connect(0)') MUST NOT be redefined in a verb identifier macro verb list. Note that the verb name 'connect' is not reserved and MAY be redefined in a verb list.

3.2.2 Mapping of the ProtocolDirID object

The protocolDirID OCTET STRING value for a particular application verb is represented by the protocolDirID value for the parent application, appended with the verb's layer identifier value.

Figure 2
ProtocolDirID Format for Verbs



The protocolDirID object is encoded as the protocolDirID value of the parent application, followed by four additional octets representing the verb layer. The verb layer value is encoded as [0.a.b.c] where 'a' is the high order byte, 'b' is the middle order byte, and 'c' is the low order byte of the <verbEnum> field for the specific application verb value. A valid PI verb enumeration will be encoded in the range "0.0.0.0" to "0.255.255.255", where the special value "0.0.0.0" is reserved for the implicitly defined 'connect(0)' verb.

3.2.3 Mapping of the ProtocolDirParameters object

The protocolDirParameters OCTET STRING value for a particular application verb is represented by the protocolDirParameters value for the parent application, appended with one octet containing the value zero. Although not actually used, this field is included to conform to the encoding rules defined in the Protocol Identifiers Reference [RFC2895].

3.2.4 Mapping of the ProtocolDirLocalIndex object

The agent MUST assign an appropriate protocolDirLocalIndex value for each application verb according to the encoding rules defined for this object in [RFC2021] and [RFC2895].

3.2.5 Mapping of the protocolDirDescr object

The agent MUST convey the <verbName> value for a particular application verb in the protocolDirDescr object. This object SHOULD be encoded as the protocolDirDescr value for the parent application appended with a 'dot' character, followed by the exact text contained in the <verbName> field.

3.2.6 Mapping of the protocolDirType object

The agent MUST set the protocolDirType object for each application verb to the value representing the empty bit set ({}).

3.2.7 Mapping of the protocolDirAddressMapConfig object

The agent MUST set the protocolDirAddressMapConfig object for each application verb to the value 'notSupported(1)'.

3.2.8 Mapping of the protocolDirHostConfig object

The agent MUST set the protocolDirHostConfig object for each application verb present in the protocol directory according to the monitoring capabilities for each verb. The agent MAY set this object to the same value as configured in the parent application protocolDirHostConfig object. The agent MAY choose to transition this object from the value 'supportedOn(2)' to 'supportedOff(3)' if the parent application protocolDirHostConfig object first transitions from 'supportedOn(2)' to 'supportedOff(3)'.

3.2.9 Mapping of the protocolDirMatrixConfig object

The agent MUST set the protocolDirMatrixConfig object for each application verb according to the monitoring capabilities for each verb. The agent MAY set this object to the same value as configured in the parent application protocolDirMatrixConfig object. The agent MAY choose to transition this object from the value 'supportedOn(2)' to 'supportedOff(3)' if the parent application protocolDirMatrixConfig object first transitions from 'supportedOn(2)' to 'supportedOff(3)'.

3.2.10 Mapping of the protocolDirOwner object

This object is encoded exactly the same for application verbs as for other protocolDirTable entries, according to the rules specified in the RMON-2 MIB [RFC2021].

3.2.11 Mapping of the protocolDirStatus object

This object is encoded exactly the same for application verbs as for other protocolDirTable entries, according to the rules specified in RMON-2 MIB [RFC2021].

4. Implementation Considerations

This section discusses the implementation implications for agents which support verbs in the protocol directory and the RMON collections which utilize the protocol directory.

4.1 Stateful Protocol Decoding

Implementations of the RMON-2 MIB for application layer and network layer protocols typically require little if any state to be maintained by the probe. The probe can generally decide whether to count a packet and its octets on the packet's own merits, without referencing or updating any state information.

Implementations of the RMON-2 MIB at the verb layer will, for many protocols, need to maintain state information in order to correctly classify a packet as "belonging" to one verb or another. The examples below illustrate this point.

For SNMP over UDP, a Response-PDU for an SNMP Get-PDU can't be distinguished from a Response-PDU for a Getnext-PDU. A probe would need to maintain state information in order to correlate a Response-PDU from B to A with a previous request from A to B.

For application protocols carried over a stream-based transport such as TCP, the information required to identify an application verb can span several packets. A probe would need to follow the transport-layer flow in order to correctly parse the application-layer data.

4.2 Packet Capture

For packet capture based on verb-layer protocol directory filtering, the decision to include a packet in the capture buffer may need to be deferred until the packet can be conclusively attributed to a

particular verb. A probe may need to pre-buffer packets while deciding to include or exclude them from capture based on other packets that have not yet arrived.

4.3 RMON-2 MIB Collections

Data collections such as the protocol distribution or Application Layer Host Table (alHostTable) require that each packet is counted only once, i.e., a given packet is fully classified as a single protocol encapsulation which resolves to a single leaf entry in the protocol directory. Also, octet counters related to protocol classification are incremented by the entire size of packet, not just the octets associated with a particular encapsulation layer.

It is possible that particular application protocols will allow multiple types of verbs to be present in a single packet. In this case, the agent MUST choose one verb type, and therefore one protocol directory entry, in order to properly count such a packet.

It is an implementation-specific matter as to which verb type an agent selects to identify a packet in the event more than one verb type is present in that packet. Some possible choices include:

- the first verb type encountered in the packet
- the verb type with the most instances in the packet
- the verb type using the largest number of octets in the packet
- the most 'interesting' verb type in the packet (based on knowledge of that application protocol).

5. Intellectual Property

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

6. Acknowledgements

This memo is a product of the RMONMIB WG.

7. Normative References

[RFC1905] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1905, January 1996.

[RFC1906] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, January 1996.

[RFC2021] Waldbusser, S., "Remote Network Monitoring MIB (RMON-2)", RFC 2021, January 1997.

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2571] Harrington, D., Presuhn, R. and B. Wijnen, "An Architecture for Describing SNMP Management Frameworks", RFC 2571, April 1999.

[RFC2572] Case, J., Harrington D., Presuhn R. and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", RFC 2572, April 1999.

[RFC2573] Levi, D., Meyer, P. and B. Stewart, "SNMPv3 Applications", RFC 2573, April 1999.

[RFC2574] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", RFC 2574, April 1999.

[RFC2575] Wijnen, B., Presuhn, R. and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", RFC 2575, April 1999.

[RFC2578] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[RFC2579] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC2580] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[RFC2895] Bierman, A., Bucci, C. and R. Iddon, "Remote Network Monitoring MIB Protocol Identifiers", RFC 2895, August 2000.

8. Informative References

[RFC1155] Rose, M. and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", STD 16, RFC 1155, May 1990.

[RFC1157] Case, J., Fedor, M., Schoffstall, M. and J. Davin, "Simple Network Management Protocol", STD 15, RFC 1157, May 1990.

[RFC1212] Rose, M. and K. McCloghrie, "Concise MIB Definitions", STD 16, RFC 1212, March 1991.

[RFC1215] Rose, M., "A Convention for Defining Traps for use with the SNMP", RFC 1215, March 1991.

[RFC1901] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Introduction to Community-based SNMPv2", RFC 1901, January 1996.

[RFC2570] Case, J., Mundy, R., Partain, D. and B. Stewart, "Introduction to Version 3 of the Internet-standard Network Management Framework", RFC 2570, April 1999.

9. IANA Considerations

At this time there are no application protocol verbs defined that require IANA registration, similar to the 'ianaAssigned' protocol identifiers found in RFC 2895. It is remotely possible that a future version of this document will contain application verb definitions which require assignment in the 'ianaAssigned' protocol identifier subtree.

10. Security Considerations

This memo defines the structure of a portion of the Remote Monitoring MIB framework, but does not define any MIB objects or protocol operations. Instead, it defines algorithms for representing application protocol verbs in RMON Protocol Identifiers. It does not introduce any new security risks into a managed system.

However, if an MIB collection is designed which utilizes this type of Protocol Identifier, then such a collection may expose which verbs in an application protocol are used in a network. Inclusion of this additional information may require more consideration for protection. MIB writers should address such considerations.

Appendix A: Usage Examples

The following examples are listed to demonstrate how RMON verb identifiers are declared.

A.1 FTP Example

This example defines verb enumeration values for the File Transfer Protocol as defined in RFC 959 and updated by RFC 2228 and RFC 2640. Note that verb name strings specified in the <verbName> field are not limited to 4 characters in length. In the FTP protocol, all the command names are 4 characters in length and the verb name string should match the official command name as closely as possible.

```
ftp VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for FTP is derived from the list
       of commands defined for the File Transfer Protocol,
       which are identified by case-insensitive strings.
       The commands are simply listed in the order found
       in the FTP documentation."
    REFERENCE
      "File Transfer Protocol, RFC 959, Section 4.1;
       FTP Security Extensions, RFC 2228, Section 3;
       Internationalization of the File Transfer Protocol,
       RFC 2640, Section 4.1."
    ::= {
          user(1),     -- USER NAME
          pass(2),     -- PASSWORD
          acct(3),     -- ACCOUNT
          cwd(4),      -- CHANGE WORKING DIRECTORY
          cdup(5),     -- CHANGE TO PARENT DIRECTORY
          smnt(6),     -- STRUCTURE MOUNT
          rein(7),     -- REINITIALIZE
          quit(8),     -- LOGOUT
          port(9),     -- DATA PORT
          pasv(10),    -- PASSIVE
          type(11),    -- REPRESENTATION TYPE
          stru(12),    -- FILE STRUCTURE
          mode(13),    -- TRANSFER MODE
          retr(14),    -- RETRIEVE
          stor(15),    -- STORE
          stou(16),    -- STORE UNIQUE
          appe(17),    -- APPEND (with create)
          allo(18),    -- ALLOCATE
          rest(19),    -- RESTART
          rnfr(20),    -- RENAME FROM
          rnto(21),    -- RENAME TO
```

```
            abor(22),    -- ABORT
            dele(23),    -- DELETE
            rmd(24),     -- REMOVE DIRECTORY
            mkd(25),     -- MAKE DIRECTORY
            pwd(26),     -- PRINT WORKING DIRECTORY
            list(27),    -- LIST
            nlst(28),    -- NAME LIST
            site(29),    -- SITE PARAMETERS
            syst(30),    -- SYSTEM
            stat(31),    -- STATUS
            help(32),    -- HELP
            noop(33),    -- NOOP
            auth(34),    -- AUTHENTICATION/SECURITY MECHANISM
            adat(35),    -- AUTHENTICATION/SECURITY DATA
            pbsz(36),    -- PROTECTION BUFFER SIZE
            prot(37),    -- DATA CHANNEL PROTECTION LEVEL
            ccc(38),     -- CLEAR COMMAND CHANNEL
            mic(39),     -- INTEGRITY PROTECTED COMMAND
            conf(40),    -- CONFIDENTIALITY PROTECTED COMMAND
            enc(41),     -- PRIVACY PROTECTED COMMAND
            lang(42)     -- LANGUAGE
    }
```

A.2 POP3 Example

This example defines verb enumeration values for the Post Office Protocol, Version 3, as defined in RFC 1939 and updated by RFC 2449.

```
pop3 VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for POP3 is derived from the list
       of commands defined for the Post Office Protocol,
       which are identified by case-insensitive strings.
       The commands are simply listed in the order found
       in the POP3 command summary."
    REFERENCE
      "Post Office Protocol, Version 3, RFC 1939, Section 9;
       POP3 Extension Mechanism, RFC 2449, Section 5."
    ::= {
          user(1),
          pass(2),
          quit(3),
          stat(4),
          list(5),
          retr(6),
          dele(7),
          noop(8),
          rset(9),
```

```
            apop(10),
            top(11),
            uidl(12),
            capa(13)
        }
```

A.3 SNMP Example

This example defines verb enumeration values for the Simple Network Management Protocol, as defined in RFC 1905.

```
snmp VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for SNMP is derived from the list
       of PDU transaction types in the Protocol Operations
       document for SNMPv2.  Note that the 'Response'
       and 'Report' PDUs are not considered verbs, but are
       classified as belonging to the transaction type
       associated with the request PDU."
    REFERENCE
      "Protocol Operations for Version 2 of the
       Simple Network Management Protocol (SNMPv2),
       RFC 1905, Section 3."
    ::= {
          get(1),
          get-next(2),
          get-bulk(3),
          set(4),
          inform-request(5),
          trap(6)
    }
```

A.4 HTTP Example

This example defines verb enumeration values for the Hypertext Transfer Protocol, version 1.1, as defined in RFC 2616.

```
http VERB-IDENTIFIER
    DESCRIPTION
      "The set of verbs for HTTP is derived from the list
       of methods defined for the Hypertext Transfer Protocol,
       which are identified by case-sensitive strings.
       The commands are simply listed in the order found
       in the HTTP/1.1 documentation.  Methods commonly used
       in HTTP/1.0 are a proper subset of those used in HTTP/1.1.
       Both versions of the protocol are in current use."
    REFERENCE
       "Hypertext Transfer Protocol -- HTTP/1.1, RFC 2616,
```

```
          Section 9; Hypertext Transfer Protocol -- HTTP/1.0, RFC
          1945, Section 8."
      ::= {
            options(1),
            get(2),
            head(3),
            post(4),
            put(5),
            delete(6),
            trace(7),
            connect(8)  -- reserved for future use by HTTP/1.1
      }
```

A.5 SMTP Example

This example defines verb enumeration values for the Simple Mail Transfer Protocol as defined in RFC 2821.

```
smtp VERB-IDENTIFIER
    DESCRIPTION
    "The set of verbs for SMTP is derived from the set of commands
     defined for the protocol.  These commands are identified
     by case-insensitive strings.  Commands are listed in the
     order found in RFC 2821.  The special "xcmd" verb is defined
     here as a catch-all for private-use commands, which must
     start with the letter 'X'."
    REFERENCE
      "Simple Mail Transfer Protocol -- RFC 2821, sections 4.1.1
       and 4.1.5."
    ::= {
          ehlo(1),  -- Extended HELLO (4.1.1.1)
          helo(2),  -- HELLO (4.1.1.1)
          mail(3),  -- MAIL (4.1.1.2)
          rcpt(4),  -- RECIPIENT (4.1.1.3)
          data(5),  -- DATA (4.1.1.4)
          rset(6),  -- RESET (4.1.1.5)
          vrfy(7),  -- VERIFY (4.1.1.6)
          expn(8),  -- EXPAND (4.1.1.7)
          help(9),  -- HELP (4.1.1.8)
          noop(10), -- NOOP (4.1.1.9)
          quit(11), -- QUIT (4.1.1.10)
          xcmd(12)  -- Catch-all for private-use "X" commands (4.1.5)
    }
```

Authors' Addresses

Andy Bierman
Cisco Systems, Inc.
170 West Tasman Dr
San Jose, CA USA 95134

Phone: +1 408-527-3711
EMail: abierman@cisco.com

Chris Bucci
Cisco Systems, Inc.
170 West Tasman Dr
San Jose, CA USA 95134

Phone: +1 408-527-5337
EMail: cbucci@cisco.com

Russell Dietz
Hifn, Inc.
750 University Ave
Los Gatos, CA, USA 95032-7695

Phone: +1 408-399-3623
EMail: rdietz@hifn.com

Albin Warth

EMail: dahoss@earthlink.net

Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.